meadowlark optics
spatial light modulators

User Manual

XY Phase Series

Spatial Light Modulators

With PCIe Controller

512 x 512 SLM **1920 x 1152 SLM**

Mead

5964 Iris Parkway **Fax: (303) 833-4335**

Frederick, CO 80530 **sales@Meadowlark.com**

USA **www.Meadowlark.com**

Table of Contents

List of Figures

1 Introduction

This User Manual covers the 1920x1152 XY Phase Series of liquid crystal on silicon spatial light modulators (SLMs) now available from Meadowlark Optics. The manual instructs users on how to set up the PCIe electronic hardware, optics, and operate the system software. SLM system setup is fairly complex, incorporating optical components, mounting and positioning hardware, custom drive electronics hardware, and a proprietary software interface. We strongly recommend that all users first read and familiarize themselves with the entire User Manual before initiating the setup and start up procedures.

1.1 Spatial Light Modulator Principles

A SLM is a device that modulates light according to a fixed spatial (pixel) pattern. The XY Phase Series SLMs convert digitized data into coherent optical information appropriate for a wide variety of applications, including beam steering, optical tweezers, diffractive optics, pulse shaping, and more. These applications require modulators that can easily and rapidly change the wavefront of a coherent beam. By combining the electro-optical performance characteristics of liquid crystal materials with silicon-based digital circuitry, Meadowlark Optics now offers high resolution SLMs that are also physically compact and optically efficient.

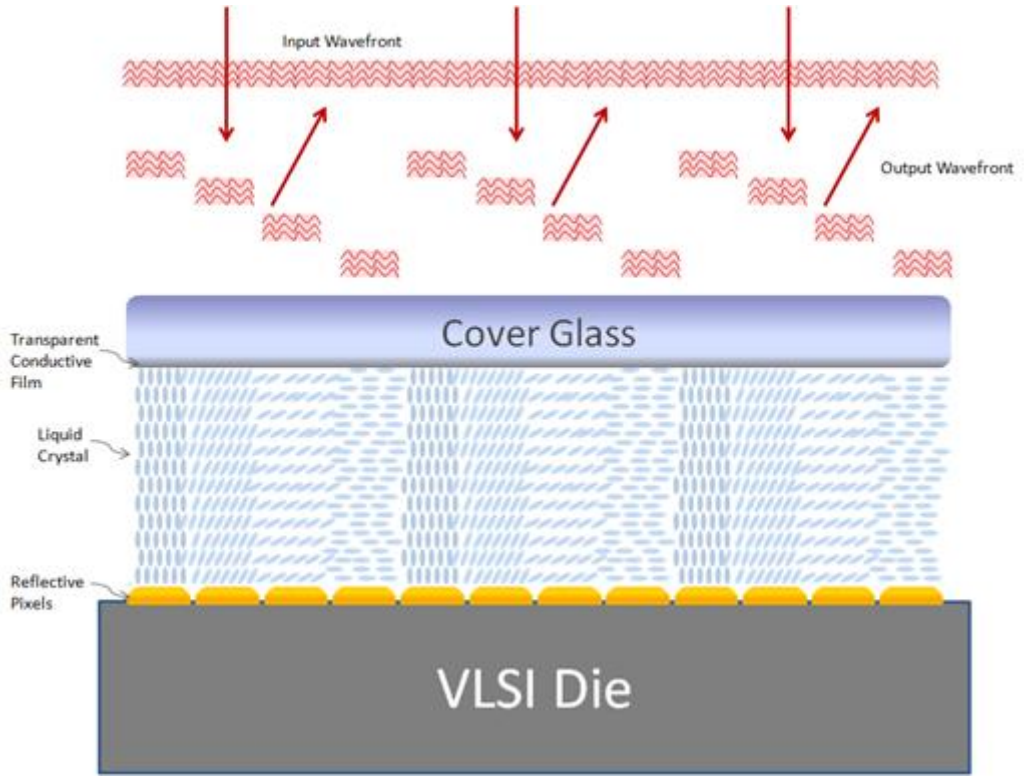


Figure 1 Cross sectional illustration of a liquid crystal spatial light modulator.

Figure 1 shows the cross section of an LC SLM. Polarized light enters the device from the top, passes through the cover glass, transparent conductive film and liquid crystal layer, is reflected off the pixel electrodes and returns on the same path. Drive signals from the HDMI interface independently drive each pixel on the SLM. The analog voltage applied to each pixel produces an electric field between that pixel and the cover glass, resulting in a change in the optical properties of the LC layer. Because each pixel is independently controlled, a phase pattern may be generated by loading different voltages onto each pixel.

1.2 Phase Modulation

The XY Phase Series of Spatial Light Modulators are fabricated with nematic liquid crystal, aligned in a homogenous configuration. Nematic liquid crystal has a variable electro-optic response to voltage. Fig. 2 shows a simplified side view of a spatial light modulator's liquid crystal layer. When no voltage is applied to the LC, the molecules are parallel to the SLM coverglass and VLSI backplane. In this case, incident light will experience the largest difference between the extraordinary (n_e) and ordinary (n_o) index of refraction. As the voltage applied to the liquid crystal is increased, the LC will tilt until the extreme is reached and the LC is nearly normal to the SLM coverglass and VLSI backplane. At this point the difference between the extraordinary and ordinary index of refraction is nearly zero.

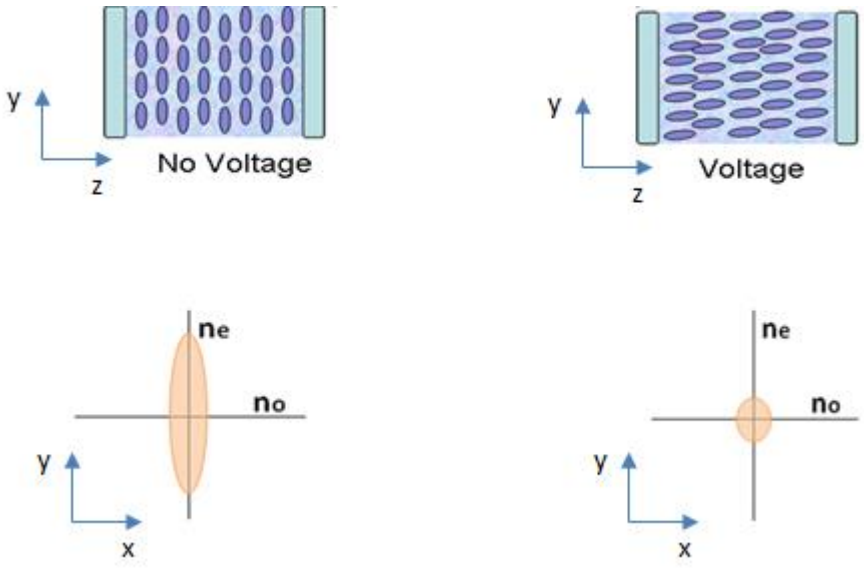


Figure 2 This diagram illustrates the liquid crystal orientation with respect to the coverglass and VLSI backplane as a function of applied voltage. The molecules are parallel to the coverglass and backplane when no field is applied, and are nearly perpendicular to the coverglass and backplane when full field is applied.

If light incident upon the SLM is linearly polarized parallel to the extraordinary axis, then a pure, voltage dependent phase shift will be observed. For example, if no voltage is applied to the pixel, the maximum phase retardation (typically a full wave at the design wavelength) will be applied. Likewise, if the pixel is programmed for maximum voltage, a minimum phase delay is applied. The result is a programmable phase modulation on a per pixel basis.

Each of the SLM pixels is independently programmable to 256 discrete voltage states, all providing phase modulation. The response of the liquid crystal within the SLM to the applied pixel value (voltage) is nonlinear. To account for this, each SLM is shipped with a custom look-up-table (LUT). When this LUT is applied to an input image, the result is a linear output phase response ranging from 0 to 2π.

1.3 Optical Test Setup

Depending on the application of the XY Phase Series SLM, many different optical setups can be used for either combined phase-amplitude mode or phase-only mode. Two examples of phase-only optical test setups are shown below.

The first optical setup, illustrated in Figure 3, is a modification of a Twyman-Green interferometer (Handbook of Optics. Vol. 1, pp. 2.28-2.29). Here a monochromatic, collimated light source (i.e. laser beam expanded so it is larger than the diagonal of the SLM) passes through a non-polarizing beamsplitter such that the beam is divided into two beams, with nearly equal intensity. One of these beams illuminates the XY Phase Series SLM, while the other illuminates a reference mirror. Each of these reflected beams is then recombined at the image plane of a lens. If the reference mirror and the SLM are carefully aligned such that they are nearly coplanar, interference fringes will be visible at the image plane. A camera is placed at the image plane in order to magnify the fringes for easier viewing. When the XY Phase Series SLM is driven with different phase patterns, dynamic interference fringes can be viewed. Analyzing the interference fringes will then provide insight into the phase modulation provided by the XY Phase SLM.

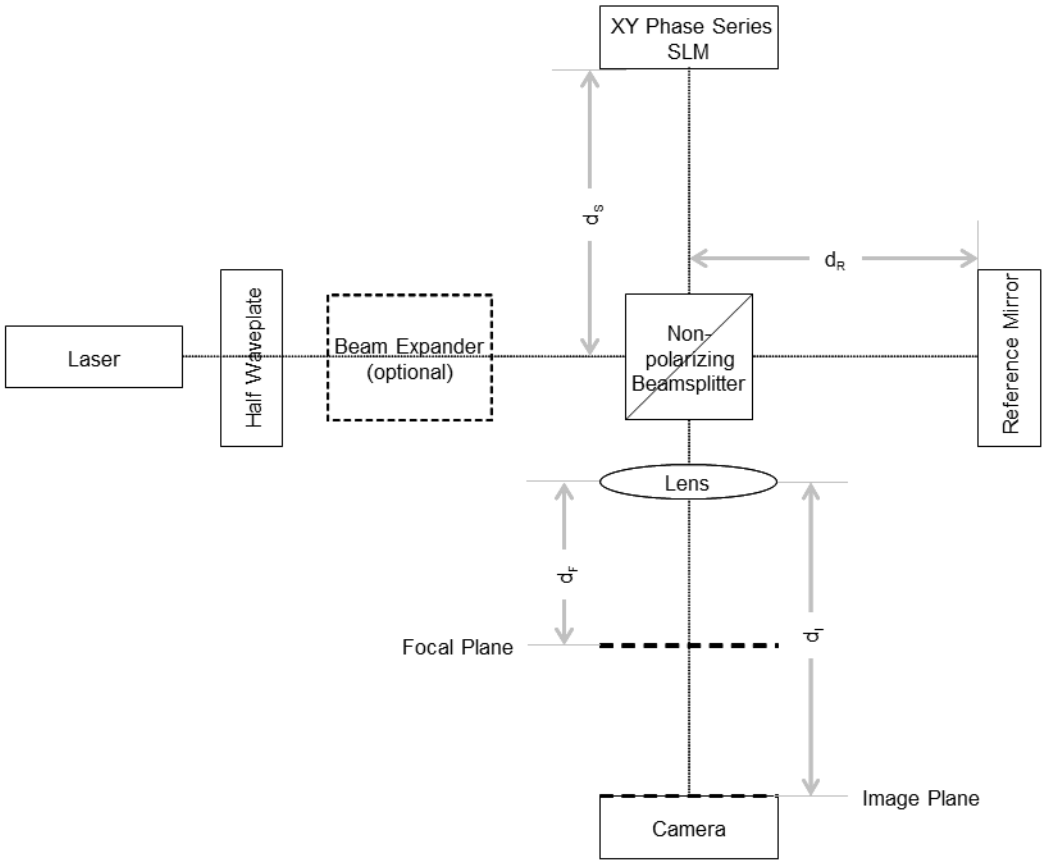


Figure 3 A Twyman-Green interferometer for testing an XY Phase SLM.

In order to ease the alignment of the reference mirror and the XY Phase Series SLM in the Twyman-Green interferometer, place the camera or a card into the beam path at the focal plane of the lens. There should be two spots visible, one from the reference mirror and one from the SLM (there may also be very faint high-order diffraction spots from the SLM). If utilizing a beamsplitter cube, there are likely to be additional spots that are ghost images from the beamsplitter. When adjusting the tip/tilt of either the reference mirror or the SLM, the ghost spots will not move, but the true spots from the reference mirror and the SLM will move. Once the correct two spots are located, adjust the tip and tilt of the reference mirror and/or the SLM until these two spots are aligned on top of one another. Place the camera back in the image plane, or remove the card, and interference fringes should be visible in the image plane. By slowly adjusting the tip and/or tilt of just the reference mirror or SLM, the desired interference fringes should be readily obtained.

If working with a laser diode it is important to note that the coherence length of laser diodes are typically much shorter than the coherence length of a laser. With a Twyman-Green interferometer it is critical that both the SLM leg and the reference mirror leg have the same optical path length, ensuring that the short coherence length of a laser diode does not significantly reduce the visibility of the interference fringes. In order to ensure equal path lengths, set the

camera such that it is focused onto the SLM – distinct features of the silicon backplane, such as individual pixels or the edge of the active area, should become sharp. Blocking the reference leg beam will likely be necessary in order to prevent crosstalk in the camera. Then move the block from the reference leg to the SLM leg and move the reference mirror closer to, or further from, the beam splitter until it is also in sharp focus on the camera. Since the reference mirror will likely not have any features with which to see the best focal point, look for focus of dust particles, or perhaps very carefully place a card onto the reference mirror such that it covers about half of the beam. The card edge can then be used as a guide to find the best focal position of the reference mirror.

There are a few important issues to remember whenever working with an SLM.

1. **Polarization** – The SLM is basically a variable single-order retardation plate, or wave plate. Like all wave plates, there is a fast axis and a slow axis. However with the XY Phase Series SLMs the index of refraction along the slow axis can be decreased electronically. When the light source is linearly polarized and parallel to the slow axis of the SLM, the result is phase-only modulation of the light source. If the orientation of the incident polarization is incorrect, there would be no modulation observed with variable voltage. Placing a half wave plate between the laser and the beamsplitter will greatly facilitate achieving the desired polarization alignment. If there is any question about the laser producing linearly polarized light, place a polarizer between the laser and the half wave plate. This will ensure that the output of the laser is linearly polarized, and the half wave plate can then be used to rotate the angle of the polarization.

2. **Diffraction** – The SLM has discrete, reflective pixel pads in order to isolate the electrical signals and allow phase patterns to be written into the SLM. As a result of these discrete pixel pads, there will be diffraction. This diffraction can easily be seen in the Focal Plane of the Lens. There will be a very bright center spot (0^{th}-order), surrounded by a grid of spots becoming progressively dimmer as they get further from the 0^{th}-order. In order to attain the maximum throughput, it is suggested to use as many of the diffracted spots, or orders, as possible. However, some applications do not allow the use of more than one order (typically the 0^{th}-order).

3. **Dispersion** – Liquid crystal wave plates are not very achromatic because the index of refraction varies as a function of wavelength. This dispersion means that a device designed to provide a 2π phase stroke at one wavelength, will provide less than 2π phase stroke at a longer wavelength, and more than 2π phase stroke at a shorter wavelength. A wavelength dependent calibration is provided with the SLM such that input graylevels of 0 to 255 linearly map to phase delays of 0 to 2π. If the wavelength is changed it will be necessary re-calibrate the SLM.

4. **Optical Quality** – Due to the very small pixel pitch of the SLM, it is important to use high quality optics. A single element lens will generally have too much spherical aberration to

provide a good, sharp focus across the entire clear aperture of the SLM. As a result, it is recommended to always use at least a doublet lens. For the same reason, the longer the focal length of the lens, the better the resulting image at the image plane. In addition, the transmitted wavefront distortion of most beamsplitter cubes is typically ~λ/4, contributing to unacceptable aberrations at the image plane. The use of a beamsplitter plate in place of a beamsplitter cube could reduce aberrations, but using a beamsplitter plate requires the use of a compensating plate in the reference leg of the Twyman-Green interferometer.

An off-axis setup, shown in Figure 4, is designed to maximize throughput by eliminating the non-polarizing beam splitter. A laser beam is incident on the SLM at a slight angle, reflects off of the reflective pixel pads, and then is imaged with a lens onto a camera. Please note that because this optical setup is not an interferometer, the actual phase modulation will not be visible on the camera. This optical setup is only shown to illustrate the concept of an off-axis system. The setup should be modified to meet the exact application requirements. The off-axis angle should be kept to a minimum in order to reduce crosstalk effects due to the beam traveling through more than one pixel region. Minimizing the off-axis angle also keeps the phase stroke closer to the designed value.

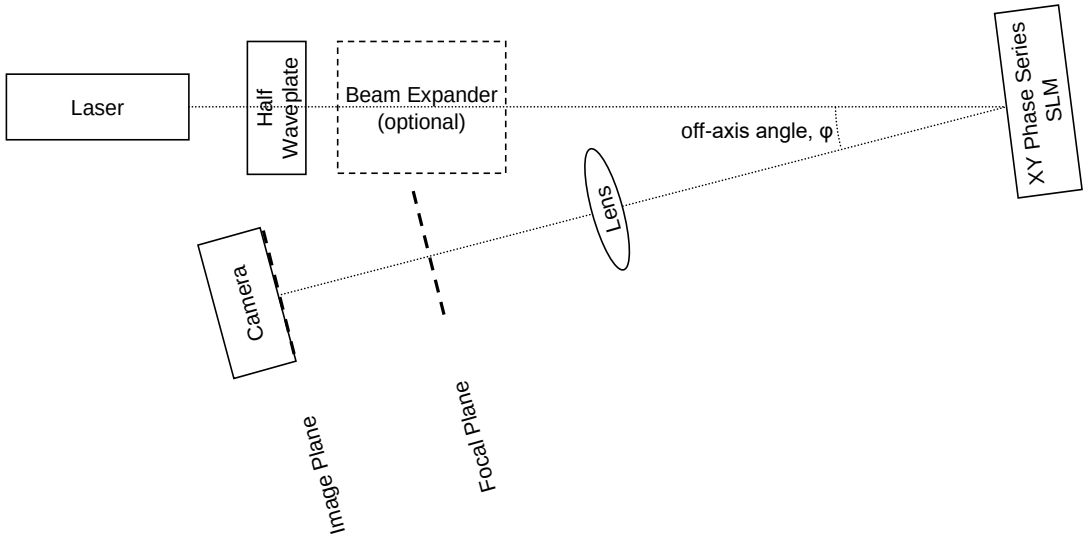


Figure 4 Off-axis optical setup for the XY Phase Series SLMs.

2 1920 x 1152 Spatial Light Modulator System Shipping Contents

The complete 1920 x 1152 SLM system, which is driven through a x8 PCIe interface, is shown in Figure 5. This system consists of a PCIe controller, the optical head containing the SLM, an external power cable, a x8 PCIe cable, a x8 PCIe host adapter card, and a USB card that contains software, documentation, and user manual. The power cable must be connected from the controller to a power strip, the PCIe cable must be connected from the host adapter card which is mounted in a x8 PCIe slot in the computer to the PCIe Driver.

Figure 5 Complete 1920x1152 SLM system with all associated cabling.

3 512 x 512 Spatial Light Modulator Shipping Contents

Figure 6 displays all the components that are contained within the PCIe system. Included in the system are the SLM optical head, the PCIe Driver box, the PCIe host interface board, a 3 ft ribbon cable, a PCIe x4 cable, an external power supply, a manual, and a software USB flash drive.

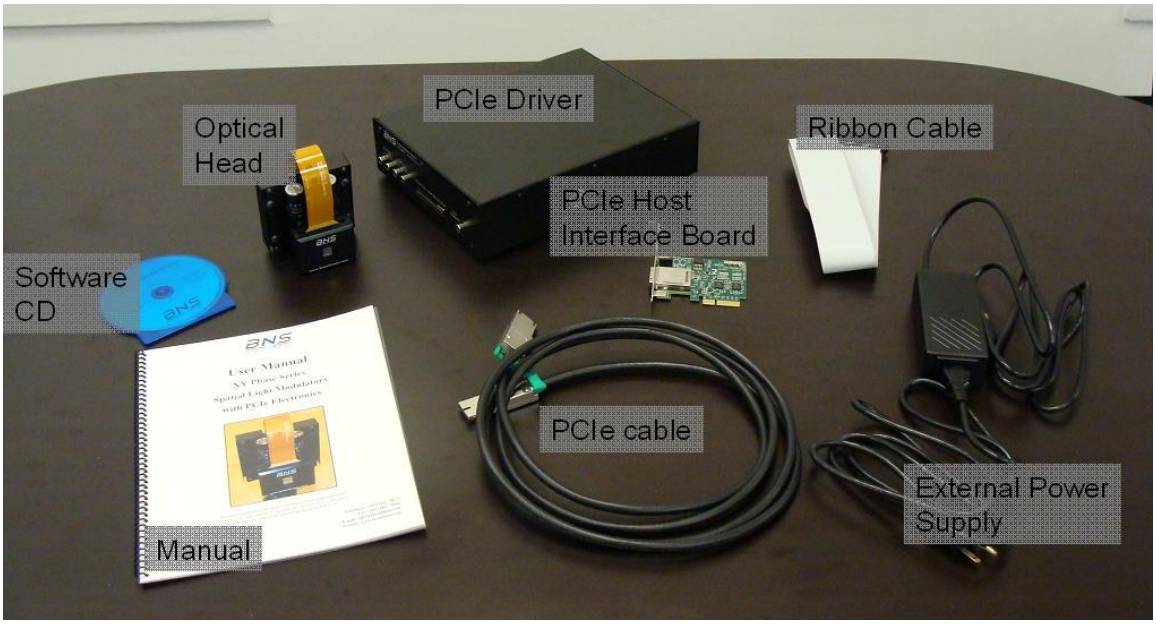


Figure 6 Complete 512x512 SLM system with all associated cabling.

4 System Setup

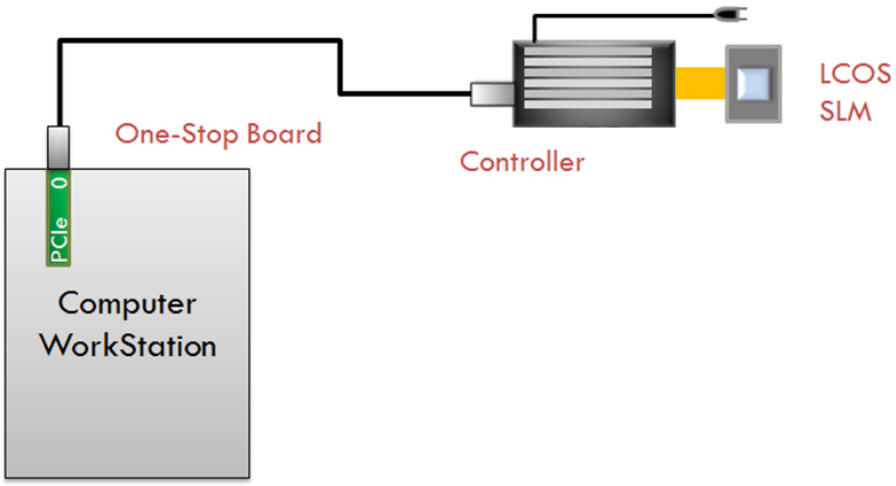


Figure 7 Electrical connection diagram.

Figure 7 shows the cabling required for the electrical system. It is assumed that the customer has a computer workstation with a free PCIe x8 slot available for the 1920x1152 SLM, or a PCIe x4 slot available for the 512x512 SLM. The computer should be running Windows® 7 or Windows® 10. It is also useful to have a small, surge protected power strip available. To setup the electrical system:

1) Power down the computer workstation

2) Open the computer and install the OneStop board in a spare PCIe slot on the computer's motherboard. Make sure the board is seated properly in the slot, and secure the board's panel bracket to the computer with a screw.

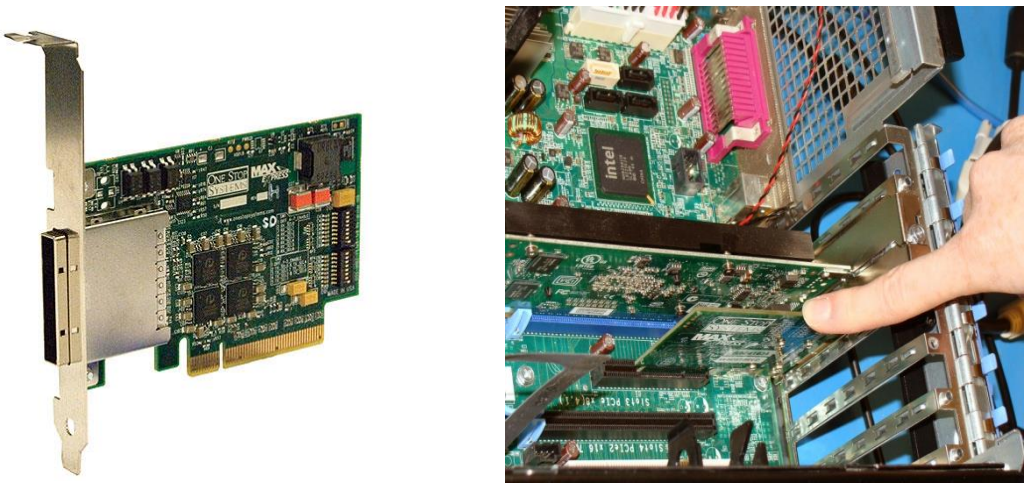
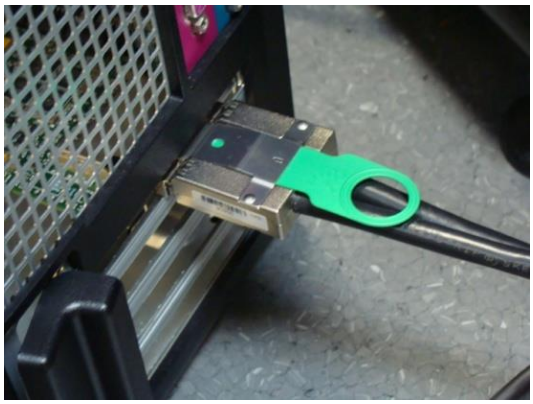


Figure 8 The One Stop Board (left) and its installation into a computer workstation (right).

3) Connect the PCIe cables between the controller box and the One-Stop Board in the Computer Workstation.



4) Connect the power cable to the controller box. Make sure the power switch is in the OFF position. Plug the power cable into the power strip. Turn on the power strip, and turn on power on the PCIe controller.
5) For the 512x512 SLM connect the Ribbon Cable from the PCIe Driver to the SLM Optical Head.
6) Boot the computer.

4.1 Minimum System Requirements

In order to effectively utilize your SLM system some basic computing hardware is required. The following components are essential to properly achieve the full performance of your SLM system.

- 1 GHz dual-core processor or better
- 2 GB RAM or better
- Available x8 PCIe slot for the 1920x1152 SLM, or x8 PCIe slot for the 512x512 SLM.
- 500 MB available disk space
- Windows®-based computer operating system (Windows 10® or Windows 7® (64-bit))
- NVidia GPU supporting OpenCL 1.2 or higher is required

4.2 Software Installation Instructions

The Meadowlark software on the USB flash drive contains the executable code necessary to operate the SLM, as well as sample pattern files, and various other support files. This software is used to load images to the SLM via the PCIe interface. Please follow these steps to install the software.

1. Insert the USB flash drive that came with your SLM system into the USB drive.

2. Double click the setup installation file

3. When the Install shield window appears, follow the onscreen instructions to complete the installation.

4. When the installation is complete, eject the USB drive from the drive and store it away carefully.

5. After installation is complete, the software should be ready to use. A shortcut to the executable will be installed on the desktop, and in the Start Menu.

6. The example program will be installed under C:\Program Files\Meadowlark Optics\Blink Overdrive Plus. The software development kit documentation and example programs can be found at C:\Program Files\Meadowlark Optics\Blink Overdrive Plus\SDK.

4.3 Device Driver Installation

Meadowlark has created a device driver for the PCIe controller. If there seems to be a problem with the system, verifying that the device drivers are installed is a good place to start. Open Device Manager and look for “Jungo” as shown in Figure 9. There should be two sub-items in this category. For the 1920x1152 SLM the SLM driver should show "SLM P1920", or for the 512x512 SLM the SLM driver should show “PCIeDriver”. Additionally a generic driver called "WinDriver1230" should be listed. When both sub-items are installed this indicates that the One Stop PCIe host adapter board is in the computer and is functioning properly.

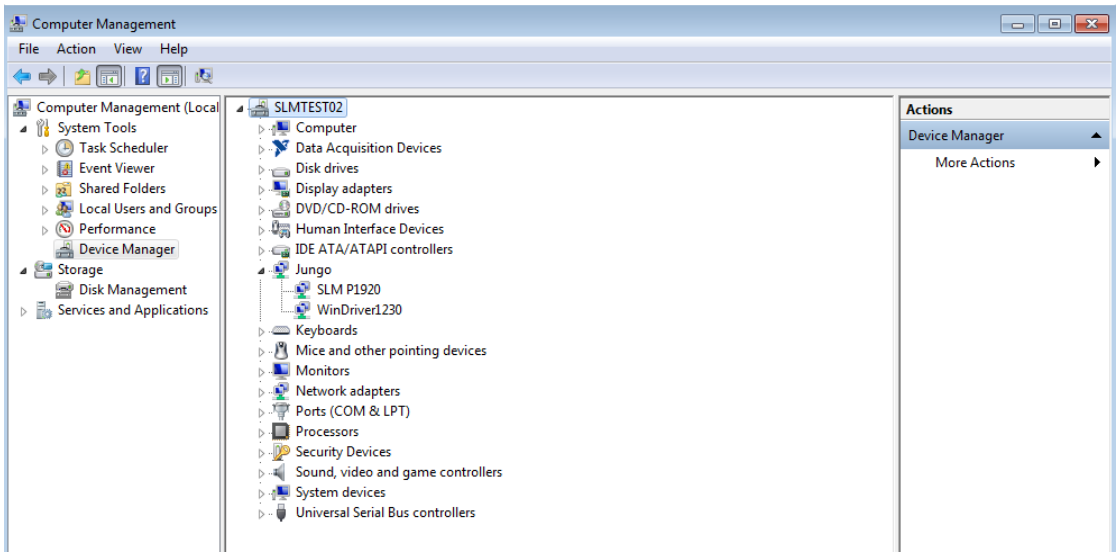


Figure 9 The Device Manager when the software is properly installed.

Note: The PCIe SLM driver will only be visible after installing the hardware and software and rebooting the computer. If, after rebooting, the drivers still do not appear, look under System

Devices for a PCI Standard RAM Controller or under Other Devices for a PCI Memory Device as shown in Figure 10.

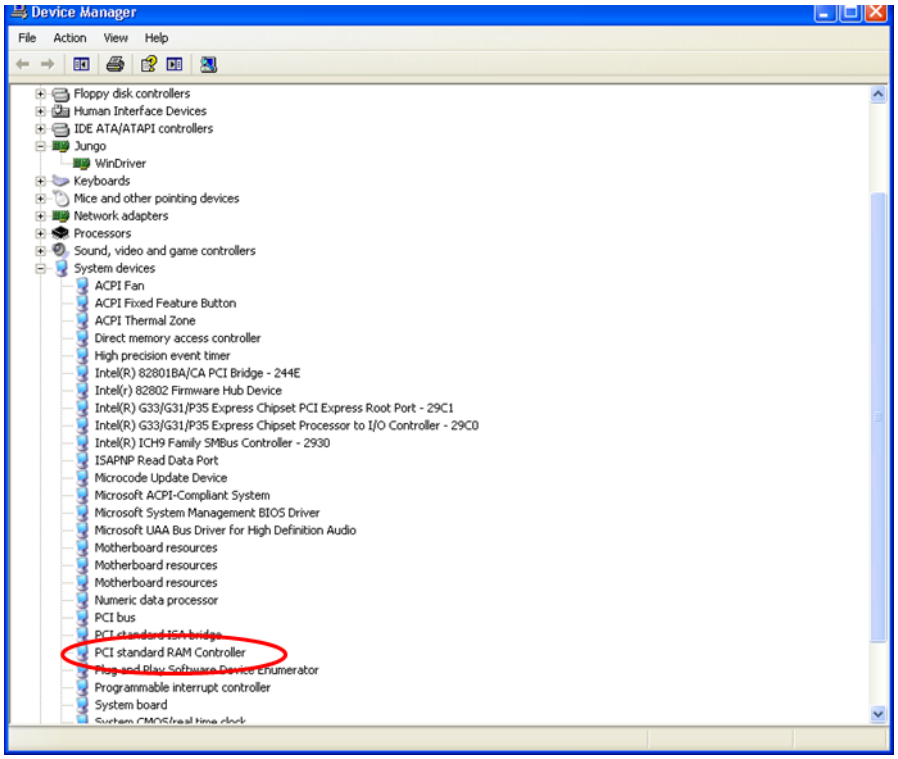


Figure 10 Device Manager prior to rebooting

If rebooting does not update the 'PCI Standard RAM Controller' or 'PCI Memory Device' driver, then right click on this item and select to update the driver. In the Hardware Update Wizard at the prompt that says 'Can Windows connect to Windows Update to search for software', select 'No, not this time'. At the next window, select to 'Install from a list or specific location'. At the next window, select 'Don't search. I will choose the driver to install'. At the next window within the Model list box the PCIe SLM option should be found. Clink on it, and proceed with the driver update. If it is not found, click on the background of the list box to un-select all items, and then click on the 'Have Disk' button. This will allow you to browse to C:\Program Files\Meadowlark Optics\Blink OverDrive Plus. If this also fails, then contact Meadowlark support for further information.

5 Software Operation

To start the software, double click the "BlinkPlus" shortcut located on your desktop, or click on the Windows "Start" button, select, "Run…", type "C:\Program Files\Meadowlark Optics\Blink Overdrive Plus\BlinkPlus.exe", then click "OK". When the software is started, the screen shown in Figure 11 should be visible.

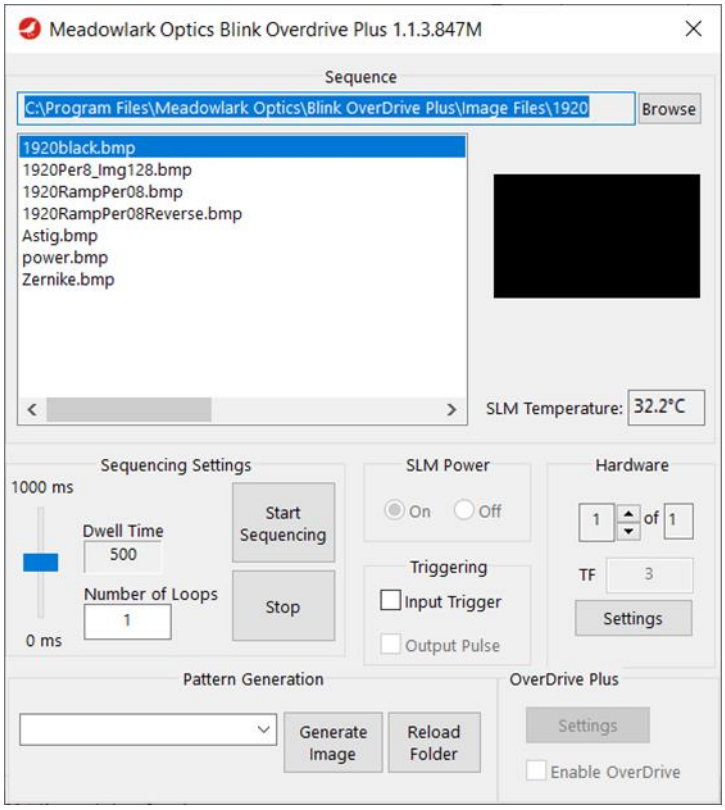


Figure 11 Main window for the Meadowlark Blink PCIe software

The SLM will be powered automatically when the computer boots. After the software has opened, the Pattern File selected in the Sequence List should appear both on the GUI and on the SLM.

5.1 Sequences

The Sequence list is defined by reading the contents of a folder and searching for bitmaps. The software stores information about the last folder used and loads the images from this folder upon initialization. Note that the images are not sorted, so to force a series of images to occur in a specific order it is recommended file names within a folder follow a numbering sequence such as: 000Img.bmp, 001Img.bmp, 002Img.bmp…etc.

To change the Pattern File currently loaded into the SLM, simply point the mouse at a different Pattern File in the Sequence List and click. This new Pattern File should now be highlighted, and be immediately loaded into the SLM. The cursor key (↑ and ↓) can also be used to select different Pattern Files in the Sequence list.

5.2 Running a sequence

The Pattern Files can be changed repetitively by clicking the "START" button located on the bottom of the main window. Once the "START" button has been pressed, the system will sequence through all the Pattern Files in the current Sequence List at a user defined frame rate (Hz). Using the single or continuous buttons the user can select to loop through the images once, or indefinitely.

5.3 Opening new sequence list

To open a new sequence file move the cursor over the "Browse…" button and click. This opens the standard Windows wizard for opening a file or folder. The user is asked to locate a folder on the hard disk. Once located, the new sequence is automatically loaded into the system with the images currently contained within the selected folder. If the images within the folder are moved or deleted after loading into the system, it is necessary to click the "Browse…" button again and reselect the folder.

5.4 SLM Temperature

The SLM software will periodically poll the current SLM temperature. The liquid crystal will lose modulation depth as the SLM heats. If the SLM is used with a high power laser it is recommended that the SLM be calibrated at the desired operating temperature.

5.5 Settings Button

The settings button opens a dialog that allows the user to browse for a LUT calibration, browse for a wavefront correction, and adjust the coverglass voltage **ONLY** if recommended by a Meadowlark Optics Employee.

5.5.1 Look Up Table (LUT) Information

Of primary interest to the user is the ability to change the LUT, or "look up table". A LUT is used to map input image data that ranges from 0 to 255 to a set of new output values that result in a linear from 0 to 2π phase shift. The software supports several LUT file formats: *.LUT, and *.txt. In each case the LUT file is a text file. A *.LUT file contains two columns: the first column in the input graylevel value (0 – 255), and the second column in the output value (0 – 2047). A *.txt calibration file is a regional calibration. In order to ensure proper formatting this type of calibration file should only be generated using Meadowlark Optics calibration tools, which are described in greater detail in Section 6.

Each SLM system is shipped with a custom LUT designed to enable a linear phase stroke versus pixel value. This LUT will be named "slmXXXX_atYYY.blt", where XXXX corresponds to the serial number of your SLM, and YYY corresponds to the wavelength the calibration was generated at. When using this LUT, a linear progression of input 0 – 255 data should result in a linear output phase of 0 - 2π.

5.5.2 Wavefront Correction File

This is an aberration correction image that is superimposed with each user defined phase pattern by adding the two images together modulo 256, and then processing the final image through the LUT calibration. This is used to remove aberrations from the SLM, or from the SLM and optical train if calibrated in the customer's optical system.

5.5.3 Coverglass Voltage

It is generally not recommended that the user edit the coverglass voltage because improper settings could cause permanent damage to the SLM. The proper default will be set by Meadowlark Optics. However, if instructed by a Meadowlark Optics employee this value can be edited.

5.6 Triggering

The user can either control the rate at which Blink sequences using software timers, or using external triggers. If the triggering option is changed, Blink must be closed and re-initialized for the new setting to take effect. Blink will remember the last state used as the default.

For the 1920x1152 SLM the input and output triggers can be accessed through SMA connectors at the bottom of the PCIe controller. An output pulse is generated when the dual port memory bank on the PCIe hardware flips, indicating that 1.18 ms after the pulse is generated a new image will begin loading on the SLM. The falling edge of an input trigger can control when the dual port flips. An output pulse will be generated to acknowledge a received trigger, and will occur up to but not in excess of 1.18 ms of receiving an external trigger falling edge. The image on the SLM will update 1.18 ms after the output pulse is generated. This timing is illustrated in Figure 12.

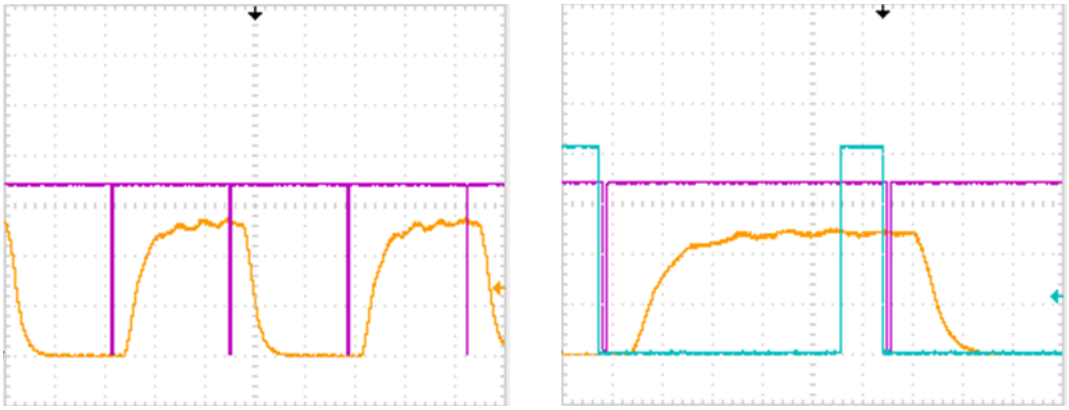


Figure 12 (left) Liquid crystal switching driven by software timing. A focal point is toggled on and off a detector (shown in yellow). When the image on the SLM changes, the hardware can produce an output pulse (shown in purple) indicating a new image will begin loading on the SLM in 1.18 ms. (right) Liquid crystal switching driven by external hardware trigger. When the falling edge of an external trigger arrives (shown in blue) the hardware will initiate the update the image on the SLM. An output pulse is generated to acknowledge receipt of the trigger (shown in purple). The image will update on the SLM 1.18 ms after the output pulse was generated (shown in yellow with the focal point moving on and off the detector).

5.7 Pattern Generation

The software allows users to dynamically generate images to load to the SLM. The supported modes include: solids, stripes, checkerboards, random phases, blazed gratings, sinusoidal gratings, Bessel beams (spiral phase, forked phase, axicons, concentric rings), holograms, Zernike polynomials, Fresnel lenses (cylindrical and circular), and superimposed images. Image generation capabilities are supported both through the main GUI, and through the software development kits by loading the Meadowlark Optics Image Generation DLL (see Section 5 for further information).

To generate an image select the image type from the Pattern Generation drop box, and then click "Generate Image". This will bring up one of the following dialogs. As the image parameters are edited, the SLM automatically updates with the current image data. The image can be saved to a user defined folder (the current folder is the default), and by clicking the "Reload Folder" button the images in the current sequence list will be automatically updated to include the newly generated image.

5.7.1 Bessel Beam

The Bessel Beam GUI (Figure 13) allows for generation of vortex beams, axicons, and concentric ring patterns. As the image parameters are edited the image written to the SLM will be updated, and the image preview will be refreshed. By default the center of the pattern will be placed at the center of the SLM. For some patterns it is easier to shift the "center" of the image applied to the SLM with pixel by pixel control rather than to optically shif the center of the illumination. Thus, the user can define the center of the image along the x and y axis.

Selecting the spiral checkbox will generate a spiral phase were "Ell" determines the number of waves of rotation in the image. Selecting a fork will superimpose a repeating phase ramp with the spiral phase, which simply shifts the vortex off axis.

Selecting Axicon will generate a linear phase ramp that is either increasing or decreasing from the center of the SLM image to the edge of the SLM. Setting the number of waves will determine the slope of the phase ramp.

Selecting Rings will generate a repeating concentric ring pattern. The user can specify the diameter in pixels of the inner ring, and the diameter in pixels of the outer ring. The pattern will consist of two graylevels defined by Value One and Value Two. In general, if using a 0 - 2π LUT, the two graylevels used will be 0 and 128 which will generate a 0 and π ring pattern. The ring pattern can be superimposed with defocus by setting the number of waves of defocus to apply.

After editing the image parameters the bitmap can be saved. The default location for saving is the current path to the current sequence list.

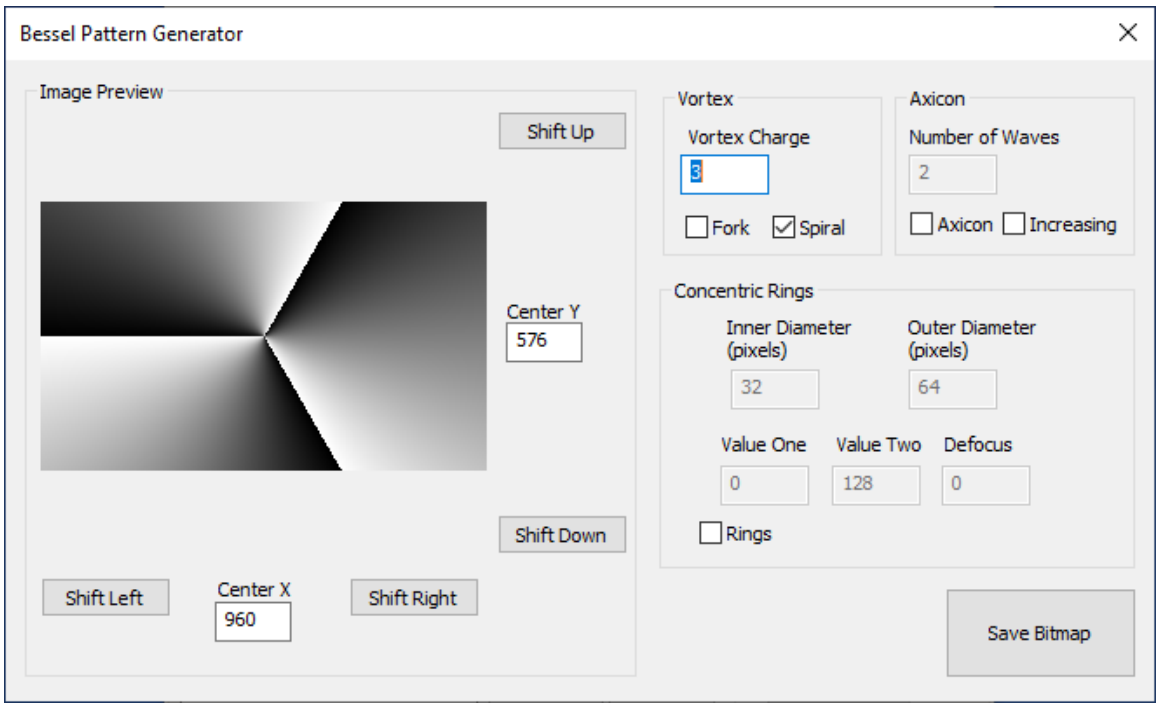


Figure 13 Bessel GUI

5.7.2 BEST Bessel Beam

The BEST Bessel Beam dialog is based on Na Ji's Bessel Scanning Technology which enables video rate volumetric imaging in two-photon microscopes. For more information on this work it is highly recommended that the following paper and corresponding supplemental information be read:

> Lu, Rongwen, Wenzhi Sun, Yajie Liang, Aaron Kerlin, Jens Bierfeld, Johannes D. Seelig, Daniel E. Wilson et al. "Video-rate volumetric functional imaging of the brain at synaptic resolution." *Nature neuroscience* 20, no. 4 (2017): 620.

To reduce the hurdles of software development when integrating volumetric imaging into a custom system or two photon microscope, Meadowlark Optics included a GUI (Figure 15) used to compute the theoretical axial point spread function (PSF) of a Bessel Beam based on the user's optical setup and the desired numerical aperture of excitation. To begin, the software requires information about the optical system, which is sketched in Figure 14. The user must define the wavelength of excitation, the beam diameter incident on the SLM, and the focal length of lens 1 which is placed between the SLM and the amplitude mask. Given the users desired effective numerical aperture for excitation (see Supplemental Information, Rongwen et. al.) the GUI will automatically calculate the intensity profile of the excitation beam at the amplitude mask. As outlined by Rongwen et. al., the amplitude mask can be designed to block all light except the 1st order peak and neighboring side lobes (Case I), or the amplitude mask can be designed to block all light except the 1st order peak (Case II). The specifications of the amplitude mask inner and outer radius requirements to support Case I or Case II are shown on the GUI, and are dynamically updated as the parameters in the GUI are adjusted.

In order to calculate the axial PSF of the excitation, it is necessary to know the manufacturer, numerical aperture (NA), magnification, and immersion type (air, water, or oil) of the objective in use. Additionally it is necessary to know the magnification of the optical relays from the amplitude mask to the galvo scanning mirror, and from the galvo scanning mirror to the objective. The optical relays should be 4F imaging systems. The magnification of the relay from the mask to the galvo is: the focal length of lens 3 divided by the focal lens 2, and the magnification of the relay from the galvo to the objective is: the focal length of lens 5 divided by the focal lens 4. If an optical relay is not found in your particular optical design, set the magnification in the software to 1. The axial PSF is automatically plotted as GUI parameters are adjusted, and the full width half max of the PSF is shown.

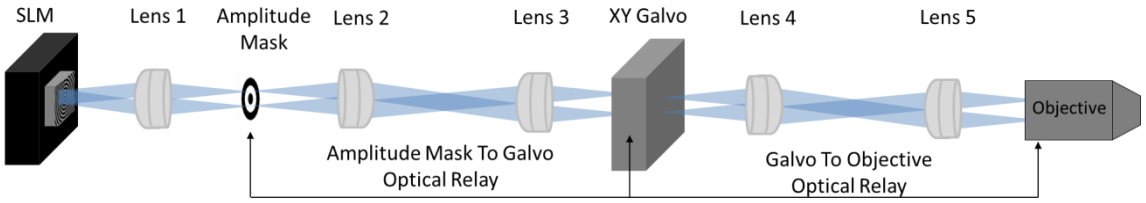


Figure 14 Optical Layout for BEST volumetric imaging

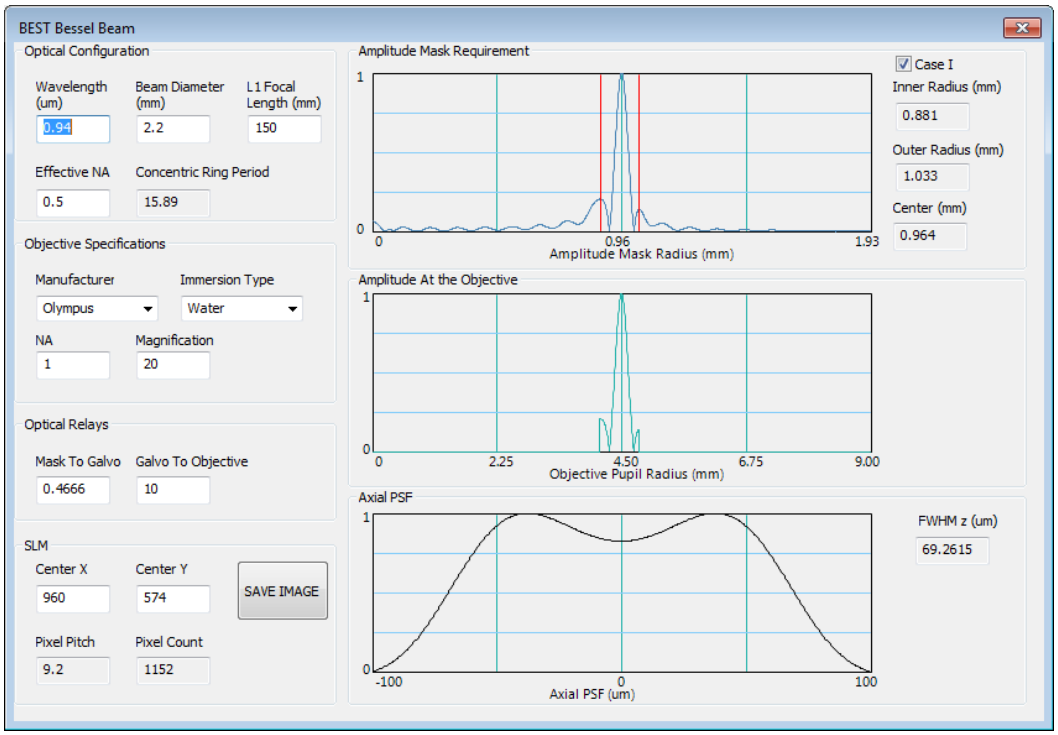


Figure 15 BEST Bessel GUI

5.7.3 Blazed or Sinusoid Grating

The blazed or sinusoidal grating GUI enables the user to create repeating ramp patterns, and sinusoidal gratings. The user must define the period in pixels from 0 - 2π, set either a horizontal

or vertical grating direction, and set the slope of the grating to either increase from 0 to 2π or decrease from 2π to 0.

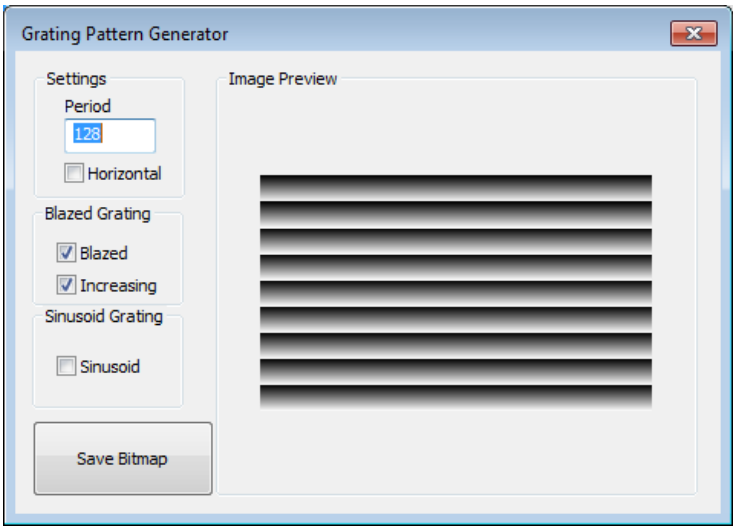


Figure 16 Grating GUI

5.7.4 Fresnel lens

The Fresnel lens GUI allows users to define either circular or cylindrical lenses. The user must define the number of pixels in the radius of the lens, and the number of waves of defocus to apply. The default center of the lens is half of the SLM height and width, but can be shifted to accommodate slight offsets in alignment.

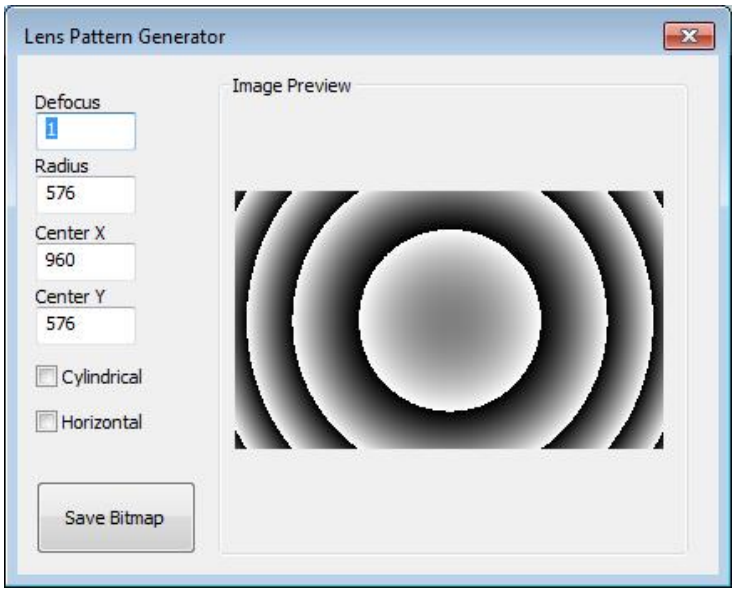


Figure 17 Fresnel Lens GUI

5.7.5 Hologram

The hologram GUI allows the user to create a phase profile that will generate a volume of focal points in the Fourier Plane. The software utilizes a Weighted Gerchberg Saxton implemented on

the GPU. This is an FFT based iterative algorithm that by default allows 10 iterations to compute a desired hologram. The user can point and click on the Target Locations to create a focal point, then manually edit the x, y, z location and intensity of the focal point. Spot locations are shown in the list box in the upper left corner. Using the arrow keys on the keyboard different spot locations can be selected for editing or removal. As the hologram is computed, a preview of the hologram is shown on the GUI, and the hologram is loaded to the SLM.

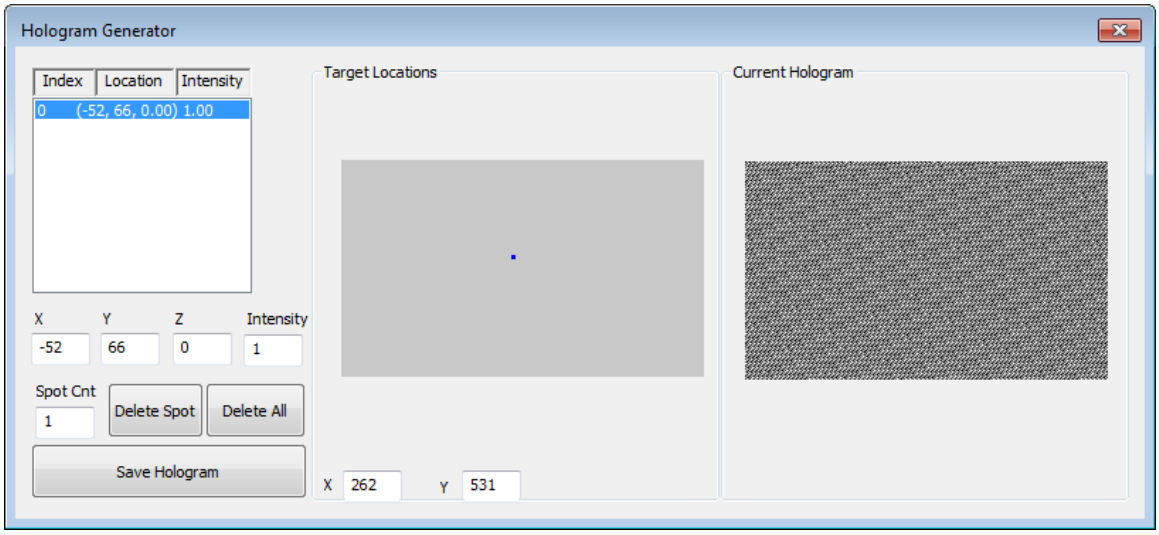


Figure 18 Hologram Generation GUI

5.7.6 Solid, Stripe, Check, or Random Phase

This GUI allows the user to create simple stripe, checkerboard, solid, or random phase patterns.

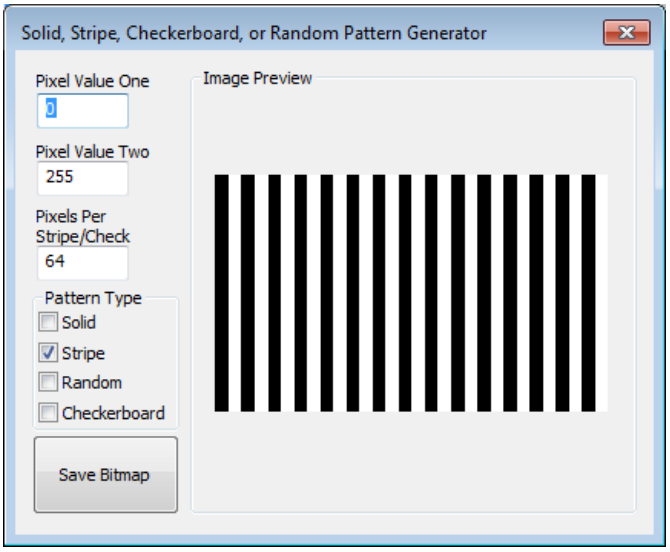


Figure 19 Solid, Stripe, Random Phase, Checkerboard GUI

5.7.7 Superimpose Images

This GUI allows the user to browse for any two images, and superimpose them by adding the two images modulo 256. The output image is then processed through the LUT calibration.

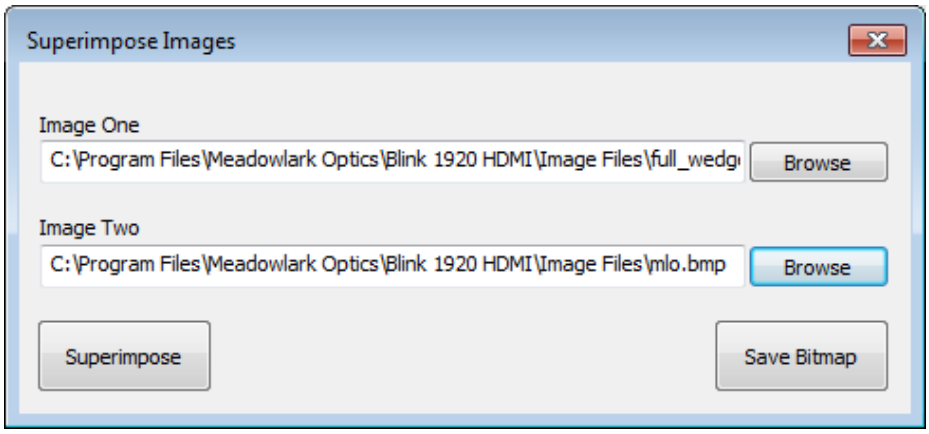


Figure 20 Superimpose Images GUI

5.7.8 Zernike Polynomials

This GUI allows the user to input Zernike polynomial coefficients and output a bitmap. The user must define a radius for the equations, the x and y location of the center of the pattern. If Zernike polynomials are being used to compute a custom aberration correction, it is recommended that the radius be a mid-point between: the distance from the center to short axis edge which tends to under-correct the corners, and the center to the diagonal which tends to over-correct the edges.

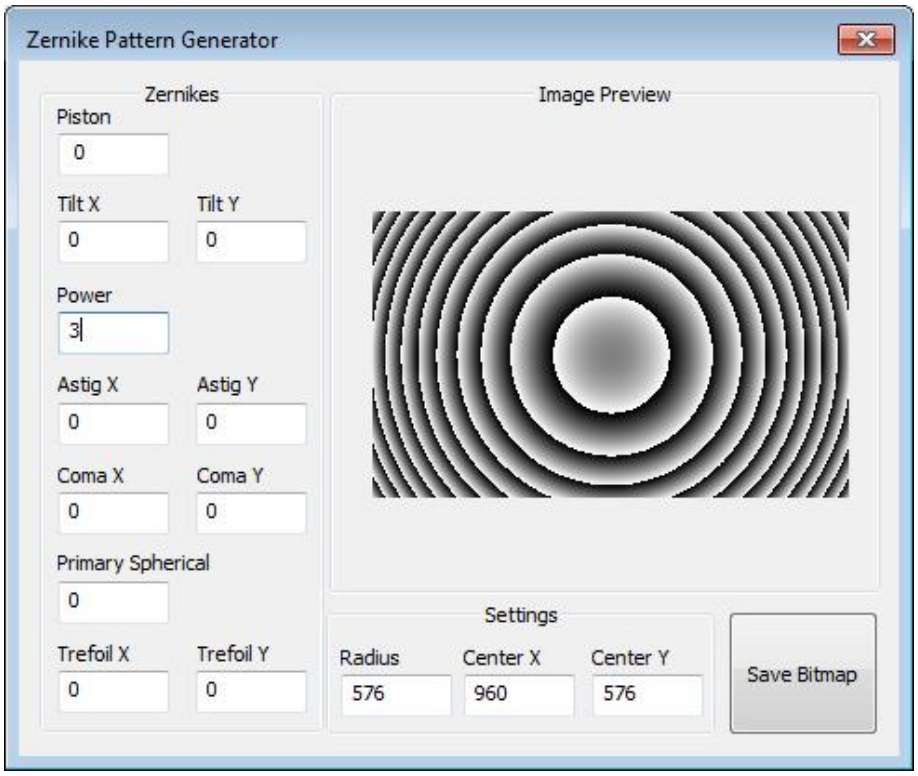


Figure 21 Zernike Polynomial GUI

6 Software Development Kit Introduction

Meadowlark Optics offers several software development kit example programs demonstrating how to interface to our SLM from Matlab, LabVIEW, and C++ with and without utilizing OverDrive Plus (ODP). Each example program demonstrates the order of operations that functions should be called in, the core functions that should be used, and how to link to our Dynamic Linked Library (DLL) to drive the Spatial Light Modulator (SLM). The purpose of this document is to outline the functions available, explain the purpose of each parameter passed, and suggest appropriate values to pass for various product options. There are two version of our DLL: Blink_SDK.dll, and Blink_SDK_C.dll. The Blink_C_wrapper DLL, and corresponding Blink_C_wrapper.h file must be used for interfacing to the SLM with Matlab and LabVIEW. This document assumes familiarity with LabVIEW, Matlab, and C++ and the Meadowlark PCIe SLM hardware.

6.1 Modes of Operation

For Matlab, LabVIEW, and C++ there are two examples demonstrating use of the SLM with and without ODP enabled. Figure 22 shows the order of function calls with green indicating initialization, blue indicating the customers loop of loading images to the SLM, and red indicating the proper shut down procedure. The function calls are further detailed in Function Summary and Usage.

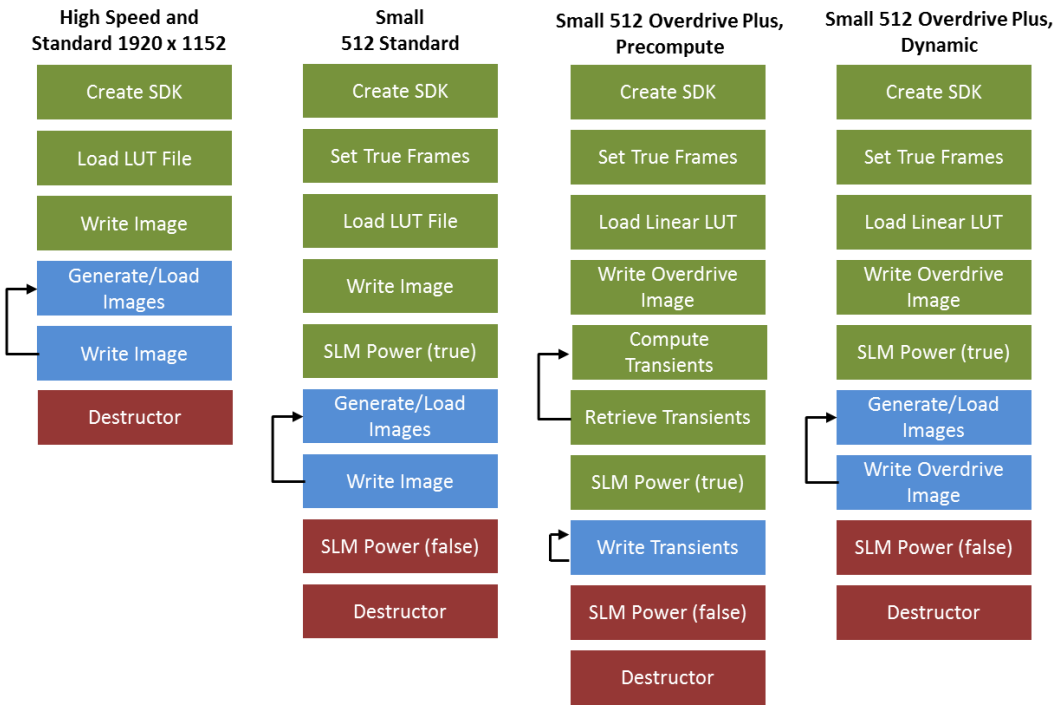


Figure 22: Function Call Order of Operations

7 Function Summary and Usage

From LabVIEW functions are accessed through Call Library Function Nodes. By double clicking on the function nodes LabVIEW allows the user to edit the function call and parameters, as shown in Figure 23.

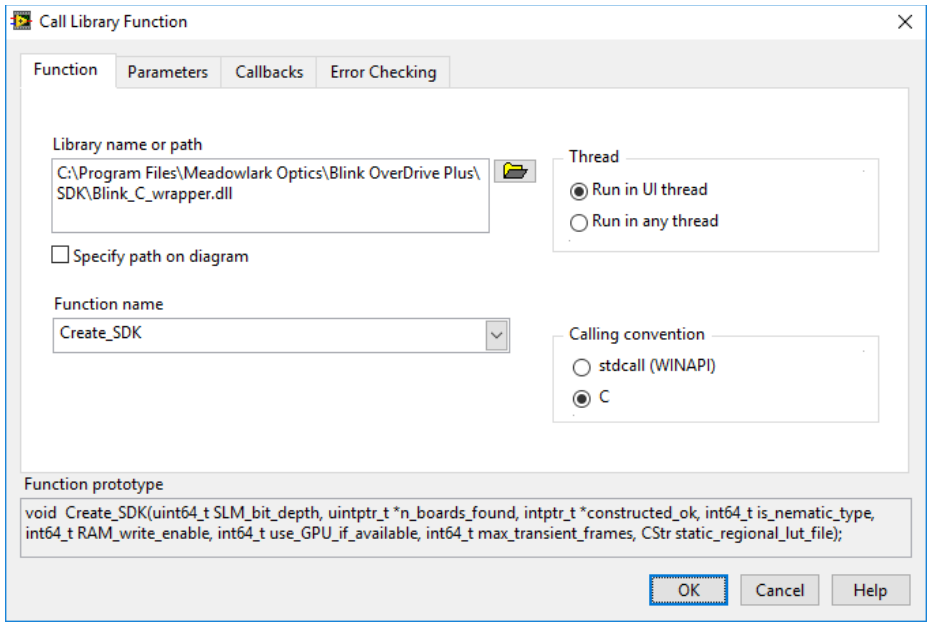


Figure 23: LabVIEW Call Library Function Node to interface to the Meadowlark Optics DLL

From Matlab the DLL is accessed by through the loadlibrary function, and subsequent calls to calllib as shown to follow. Loadlibrary takes two parameters: the Blink_C_wrapper.dll, and the .h file. The calllib function takes 'Blink_C_wrapper as the first parameter, the function name as the second parameter, and following that the function parameters as defined in the Blink_C_wrapper.h file. Note that matlab requires a C compiler. We recommend installing Microsoft SDK 7.1, then at the matlab command prompt run mex – setup to select the installed compiler.

```
if ~libisloaded('Blink_C_wrapper')
   loadlibrary('Blink_C_wrapper.dll', 'Blink_ C_wrapper.h');
end

sdk = calllib('Blink_SDK_C', 'Create_SDK', bit_depth, num_boards_found, constructed_okay,
is_nematic_type, RAM_write_enable, use_GPU, max_transients, lut_file);
```

7.1 Function Definitions

All function definitions listed below are based on the .h file that Matlab and LabVIEW can interface to.

7.1.1 void Create_SDK (unsigned int SLM_bit_depth, unsigned int n_boards_found, int *constructed_ok, int is_nematic_type, int RAM_write_enable, int use_GPU_if_available, int max_transient_frames, char* static_regional_lut_file);*

Create SDK opens communication and initializes the hardware. This is a general purpose function that supports operation with or without overdrive. Parameters that are specific to Overdrive are unused if Overdrive is not supported on the user's SLM model. The purpose, and appropriate values of the parameters passed to this function are outlined to follow.

- SLM Bit Depth –For the small 512x512 with or without ODP the bit depth should be set to 8, for the 1920x1152 the bit depth should be set to 12
- Number of Boards Found – In the constructor the hardware scans the PCIe bus for Meadowlark Optics Hardware. This parameter returns the number of boards found.
- Constructed OK – This parameter tells the user if the constructor completed successfully.
- Nematic Type – This parameter is set to 1 for SLMs built with Nematic Liquid Crystal. It is set to 0 for SLMs built with Ferroelectric Liquid Crystal.
- RAM Write Enable – Utilizing RAM writes minimizes the CPU to PCIe image transfer time. This parameter should be set to 1.
- Use GPU If Available – This parameter is specific to ODP. Computation of transient frames can be computed on the GPU if a graphics card is available. This parameter should be set to 1.
- Max Transient Frames – This parameter is specific to ODP. Multiple concepts are built into ODP to improve the LC switching speed. One concept is referred to as the transient nematic effect. As shown in Figure 24, if phase delay of a LC modulator is changed from φ_0 to φ_1, then the LC molecules relax into the new phase following approximately an exponential curve. In order to take advantage of the transient nematic effect, one instead changes the phase from φ_0 to φ_{max} to φ_1. This causes the relaxation to follow a different exponential curve until the desired phase is reached, resulting in a significantly improved switching speed. When switching between images with pixels set to arbitrary phases, the duration of time that the pixels should be set to an intermediate phase is variable based on the initial and target phase. The transient frames allow the different pixels to be held at varying voltages for an appropriate amount of time. However, on most computers each transient frame takes approximately 490 µs to load. There is a balance between minimizing the LC response time without limiting the system frame rate. In most applications we recommend this parameter be set to 10.

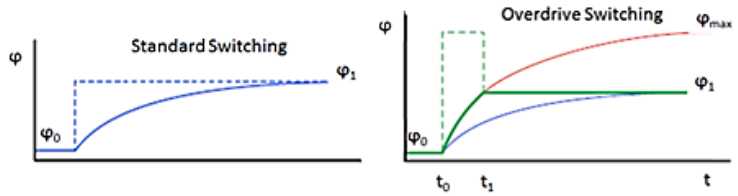


Figure 24 Standard switching, and Overdrive switching taking advantage of the transient nematic effect.

- Regional LUT – If a NULL is passed to this parameter, then ODP is disabled. Otherwise, pass the regional calibration of the LC response to applied voltage. This is a text file that was generated as a custom calibration for your SLM.

7.1.2 void Set_true_frames (int true_frames);

This function is only used by the 512 x 512 pixel SLM, and should not be called with the 1920 x 1152 pixel SLM. This function sets the DC balancing rate of the SLM. This function should be called immediately after the constructor. For Overdrive SLMs true frames should be set to 5, and or non-overdrive operation true frames should be set to 3.

7.1.3 int Load_LUT_file (int board, char LUT_file);*

This function is used to load a calibration to the hardware that corrects for the nonlinear response of the liquid crystal to voltage. If the Look-Up Table (LUT) is correct, then the user can assume that linear increments in graylevel in user defined images translate to linear increments in phase delay on the SLM. Because images are processed through the LUT in hardware, it is important that the LUT file be loaded to the hardware prior to writing images to the SLM. The function takes a board number, which should be 1 for single SLM operation and a path to the LUT file.

- 512x512 and 1920x1152 – Use this function to load the custom LUT file that shipped with the SLM to the hardware. In the example programs linear.lut is used as a default. The user should replace linear.LUT with the custom LUT delivered with their SLM.
- ODP – The regional calibration that was passed to Create_SDK is used instead of a global LUT file. The user should load a linear LUT to the hardware using this function, or the Load_linear_LUT function so that this feature is omitted.

7.1.4 int Write_image (int board, unsigned char image, unsigned int image_size, int wait_for_trigger, int external_pulse, unsigned int trigger_timeout_ms);*

This function writes an image to the SLM. The function returns when the DMA is complete, not when the hardware memory bank is ready to receive a new image. To ensure that images are never dropped you should call ImageWriteComplete after calling write image to ensure that you don't start your next DMA until the hardware is ready to receive it. The reason the write function was broken into two steps is for work with external triggers. After the DMA is complete it is safe to supply a trigger, and after ImageWriteComplete returns it is safe to start a new DMA.

Write_image function takes several parameters:

- Board – This parameter tells the software which SLM the image should be written to. For single SLM operation this should be 1.
- Image – This is a pointer to a one dimensional array containing the image data. For the 512 x 512 pixel SLM there should be 512*512 or 262,144 elements in the array, where each element is an 8-bit entry. For the 1920 x 1152 pixel SLM there should be 1920*1152, or 2,211,840 elements in the array, where each element is an 8-bit entry.
- Image Size – For the 512 x 512 pixel SLM this is the height or width of the image, which should always be 512. For the 1920 x 1152 pixel SLM this is 1920*1152.
- Wait for Trigger – This enables external triggers to control when images are loaded to the SLM. The trigger should be a TTL pulse, applied to Input A. The hardware responds to the falling edge of the trigger. If external triggers are enabled, then the function will not return until the trigger was received. If the user attempts to trigger images before an image write is complete, then the trigger will be ignored.
- External Pulse – This provides feedback to the user notifying when images are loaded to the SLM. If using external triggers this provides acknowledgement that the external trigger was received, and a new image was loaded to the SLM. This signal is normally high, and falls low for approximately 200 ns.
- Trigger Timeout – If external triggers are enabled, but they are never received by the hardware, then the software will stop waiting for a trigger when the timeout condition is met. The units of this parameter are milliseconds. In general we recommend passing 5000 for a 5 second timeout. If the timeout condition occurs the software will post an error message notifying the user of the error.

7.1.5 int ImageWriteComplete (int board, unsigned int trigger_timeout_ms);

This function checks the hardware memory bank to verify that the hardware is ready to receive a new image. Calling this function after a DMA is initiated will ensure that images are never dropped, and the SLM will never display a partial image. On the 1920x1152 SLM, if external triggers are disabled, the hardware will be ready to receive an image 1.18 ms after a DMA is complete. If external triggers are enabled, the hardware will be ready to receive an image 1.18 ms after receipt of a trigger. It is safe to supply a trigger after Write_image returns. Triggers received before a DMA is started but after ImageWriteComplete has returned will be ignored.

ImageWriteComplete takes the following parameters:

- Board – This parameter tells the software which SLM the image should be written to. For single SLM operation this should be 1.
- Trigger Timeout – If external triggers are enabled, but they are never received by the hardware, then the software will stop waiting for a trigger when the timeout condition is met. The units of this parameter are milliseconds. In general we recommend passing 5000

for a 5 second timeout. If the timeout condition occurs the software will post an error message notifying the user of the error.

7.1.6 void SLM_power (int power_state);

This function is only used for the 512 x 512 pixel SLM. This function turns the SLM power on or off. Passing 1 will turn on the SLM power, and passing 0 will turn off the SLM power. Generally it is recommended that this function be called after a valid image is loaded to the SLM.

7.1.7 void Delete_SDK ();

This function destructs the handle to the hardware, properly releases memory allocations, and shuts down the hardware. This is the last function that should be called when exiting your software.

7.2 Optional Functions

7.2.1 const char Get_last_error_message ();*

This function is optional, but is a useful check of error messages.

7.2.2 const char Get_version_info ();*

This function is optional, providing version information if requested by Meadowlark Optics.

7.2.3 int Compute_TF (float frame_rate);

This function is used to compute an appropriate DC balancing rate if using a Ferroelectric Liquid Crystal SLM.

7.2.4 double Read_SLM_temperature (int board);

This function is available with the 1920x1152. The function returns the temperature of the SLM backplane in degrees C.

7.2.5 int Get_image_height (int board);

This function returns the height of the SLM.

7.2.6 int Get_image_width (int board);

This function returns the width of the SLM.

7.3 ODP Functions – ONLY available on the 512x512 with the Overdrive Plus Upgrade

7.3.1 int Is_slm_transient_constructed ();

A true result indicates that the overdrive frame calculation engine was properly constructed, and that its required resources are available on the system.

7.3.2 int Load_linear_LUT (int board);

When using ODP the regional calibration is used to linearize the regional response of the LC to voltage. The global calibration, which is applied in hardware should be disabled by loading a linear LUT to the hardware.

7.3.3 int Write_overdrive_image (int board, unsigned char target_phase, int wait_for_trigger, int external_pulse, unsigned int trigger_timeout_ms);*

This function loads an image to the SLM using ODP. This can be called in a loop when transients are not precomputed, or it can be called to initiate a loop of precomputed images.

- Board – This parameter tells the software which SLM the image should be written to. For single SLM operation this should be 1.
- Target Phase – This tells the software the final image that the user would like to switch to. Knowing the current image, which is blank if this is called on initialization, the software computes the transient images required to minimize the LC switching time in the transition from the current image to the target phase.
- Wait for Trigger – This enables external triggers to control when images are loaded to the SLM. The trigger should be a TTL pulse, applied to Input A. The hardware responds to the falling edge of the trigger. If external triggers are enabled, then the function will not return until the trigger was received. If the user attempts to trigger images before an image write is complete, then the trigger will be ignored.
- External Pulse – This provides feedback to the user notifying when images are loaded to the SLM. If using external triggers this provides acknowledgement that the external trigger was received, and a new image was loaded to the SLM. This signal is normally high, and falls low for approximately 200 ns.
- Trigger Timeout – If external triggers are enabled, but they are never received by the hardware, then the software will stop waiting for a trigger when the timeout condition is met. The units of this parameter are milliseconds. In general we recommend passing 5000 for a 5 second timeout. If the timeout condition occurs the software will post an error message notifying the user of the error, and then will load the image to the SLM that the user attempted to write.

7.3.4 int Calculate_transient_frames (unsigned char target_phase, unsigned int* byte_count);*

This function is used to calculate the transient images between the current image, and the target image that the customer would like to transition to. This function is only used if you are pre-computing the transient images. If the customer is dynamically computing the transients, then Write_overdrive_image should be used.

- Target Phase – This tells the software the final image that the user would like to switch to. Knowing the current image, which is blank if this is called on initialization, the software

computes the transient images required to minimize the LC switching time in the transition from the current image to the target phase.

- Byte Count – This returns the size of the transient buffer required to store the transient frames

7.3.5 int Retrieve_transient_frames (unsigned char frame_buffer);*

This function is used to return the transient frames to the user so that the user can sequence through the pre-computed transient data in a loop. The user passes in a buffer to be filled, that was allocated using the byte_count retuned by Calculate_transient_frames.

7.3.6 int Write_transient_frames (int board, unsigned char frame_buffer, int wait_for_trigger, int external_puls, unsigned int trigger_timeout_ms);*

This function is used to write transient frames to the SLM. Note that the transient frames hold the target phase defined by the user Calculate_transient_frames as the last frame in the image sequence.

- Frame Buffer – This is the buffer of transient images plus the target phase as computed by Calculate_transient_frames.
- Wait for Trigger – This enables external triggers to control when images are loaded to the SLM. The trigger should be a TTL pulse, applied to Input A. The hardware responds to the falling edge of the trigger. If external triggers are enabled, then the function will not return until the trigger was received. If the user attempts to trigger images before an image write is complete, then the trigger will be ignored.
- External Pulse – This provides feedback to the user notifying when images are loaded to the SLM. If using external triggers this provides acknowledgement that the external trigger was received, and a new image was loaded to the SLM. This signal is normally high, and falls low for approximately 200 ns.
- Trigger Timeout – If external triggers are enabled, but they are never received by the hardware, then the software will stop waiting for a trigger when the timeout condition is met. The units of this parameter are milliseconds. In general we recommend passing 5000 for a 5 second timeout. If the timeout condition occurs the software will post an error message notifying the user of the error, and then will load the image to the SLM that the user attempted to write.

7.4 Optional ODP Functions – ONLY available on the 512x512 with the Overdrive Plus Upgrade

7.4.1 int Load_overdrive_LUT_file (char static_regional_lut_file);*

The regional calibration linearizes the LC response to applied voltage. This is a text file that was generated as a custom calibration for your SLM. If the file you intend to use changes from that passed to the constructor, then this function can be used to re-load the regional calibration.

7.4.2 void Stop_sequence ();

If using external triggers, and an image write has been initiated but an external trigger is not received, then the user can abort the image write using this function. In order to utilize this function it is necessary to have a multi-threaded application.

7.5 Optional Image Generation Functions

Image generation functions are accessed by including ImageGen.h and linking to ImageGen.dll or ImageGen.lib.

IMPORTANT NOTE: The code used to generate holograms and apply regional LUTs requires OpenCL to be installed on your computer, and requires a graphics card that supports OpenCL. PLEASE CHECK YOUR GRAPHICS CARD CAPABILITIES.

We recommend NVIDIA graphics cards, and use of OpenCL 1.2 or higher. To check your OpenCL version you currently have installed you can download and run GPU Caps Viewer. If you are having trouble finding this program, please contact slmsupport@meadowlark.com and we will send you a link.

7.5.1 void Generate_Stripe(unsigned char Array, int width, int height, int PixelValOne, int PixelValTwo, int PixelsPerStripe);*

This function will fill an array to define an image of user defined dimensions with a stripe pattern consisting of two pixel values, and a user specified number of pixels per stripe.

7.5.2 void Generate_Checkerboard(unsigned char Array, int width, int height, int PixelValOne, int PixelValTwo, int PixelsPerCheck);*

This function will fill an array to define an image of user defined dimensions with a checkerboard pattern consisting of two pixel values, and a user specified number of pixels per stripe.

7.5.3 void Generate_Solid(unsigned char Array, int width, int height, int PixelVal);*

This function will fill an array to define an image of user defined dimensions with a solid graylevel.

7.5.4 void Generate_Random(unsigned char Array, int width, int height);*

This function will fill an array to define an image of user defined dimensions with a random phase pattern.

7.5.5 void Generate_Zernike(unsigned char Array, int width, int height, int CenterX, int CenterY, int Radius, double Piston, double TiltX, double TiltY, double Power, double AstigX, double AstigY, double ComaX, double ComaY, double PrimarySpherical, double*

TrefoilX, double TrefoilY, double SecondaryAstigX, double SecondaryAstigY, double SecondaryComaX, double SecondaryComaY, double SecondarySpherical, double TetrafoilX, double TetrafoilY, double TertiarySpherical, double QuaternarySpherical);

This function will fill an array to define an image of user defined dimensions using Zernike polynomials. The Zernikes center is defined by center x and center y. The radius defines the number of pixels over which one wave of phase change should occur.

7.5.6 void Generate_FresnelLens(unsigned char Array, int width, int height, int CenterX, int CenterY, int Radius, double power, bool cylindrical, bool horizontal);*

This function will fill an array to define an image of user defined dimensions with a lens function. The center of the lens is defined by center x and center y. The radius defines the number of pixels over which on wave of phase change should occur. Setting cylindrical will either generate a circular or cylindrical lens, if cylindrical the lens can be horizontal or vertical.

7.5.7 void Generate_Grating(unsigned char Array, int width, int height, int Period, bool increasing, bool horizontal);*

This function will fill an array to define an image of user defined dimensions with a repeating phase ramp. The user must specify the period of the ramp, if the ramp is increasing or decreasing, and if the grating is horizontal or vertical.

7.5.8 void Generate_Sinusoid(unsigned char Array, int width, int height, int Period, bool horizontal);*

This function will fill an array to define an image of user defined dimensions with a repeating phase ramp. The user must specify the period of the ramp, and if the grating is horizontal or vertical.

7.5.9 void Generate_LG(unsigned char Array, int width, int height, int VortexCharge, int centerX, int center, bool fork);*

This function will fill an array to define an image of user defined dimensions with a spiral phase. The user should specify the vortex charge (the number of waves of phase delay in the spiral), the x and y location of the center discontinuity, and if a tilt function should be superimposed with the grating.

7.5.10 void Generate_ConcentricRings(unsigned char Array, int width, int height, int InnerDiameter, int OuterDiameter, int PixelValOne, int PixelValTwo, int centerX, int centerY);*

This function will fill an array to define an image of user defined dimensions with concentric rings to make a Bessel beam. The user should specify the inner and outer diameter of the repeating ring structure, the graylevel of the two rings, and the x and y location of the center discontinuity.

7.5.11 *void Generate_Axicon(unsigned char* Array, int width, int height, int PhaseDelay, int centerX, int centerY, bool increasing);*

This function will fill an array to define an image of user defined dimensions with a radially symmetric linear phase ramp. The user should specify the number of waves of phase delay, the x and y location of the center discontinuity, and if the phase delay is increasing or decreasing.

7.5.12 *bool Initalize_HologramGenerator(int width, int height, int iterations);*

The hologram generator relies on the GPU to complete the required computations. This function opens communication with the GPU, and specifies the height and width of the hologram that will be computed such that buffers can be allocated. This also allocates the number of iterations that will be used in computing the hologram.

7.5.13 *int CalculateAffinePolynomials(int SLM_X_0, int SLM_Y_0, int CAM_X_0, int CAM_Y_0, int SLM_X_1, int SLM_Y_1, int CAM_X_1, int CAM_Y_1, int SLM_X_2, int SLM_Y_2, int CAM_X_2, int CAM_Y_2);*

An affine transformation can be applied to the hologram computation algorithm to apply a mapping between focal point locations in the image plane, and focal point locations generated by the SLM. To calculate the transformation the user should create a series of three focal point in XY locations in SLM coordinates and record the resulting location of the 1st order focal point at a detector in the Fourier Plane. For example, Figure 25 shows one of the three measurements that must be passed as parameters to CalculateAffinePolynomials: a focal point with a SLM XY location of 128, 128, and a corresponding first order focal point on a camera. The locations will be passed as SLM_X_0 (128), SLM_Y_0 (128), CAM_X_0 (the x location of the 1st order), and CAM_Y_0 (the y location of the 1st order). To get a good calibration it is recommend that focal point locations are recorded when steering a focal point to three different quadrants. For example, SLM XY locations of: (128,128), (-128, 128), and (-128, -128). If 128 is too close to the 0th order to get an accurate measurement a wider angle can be used, such as 256 or 512.

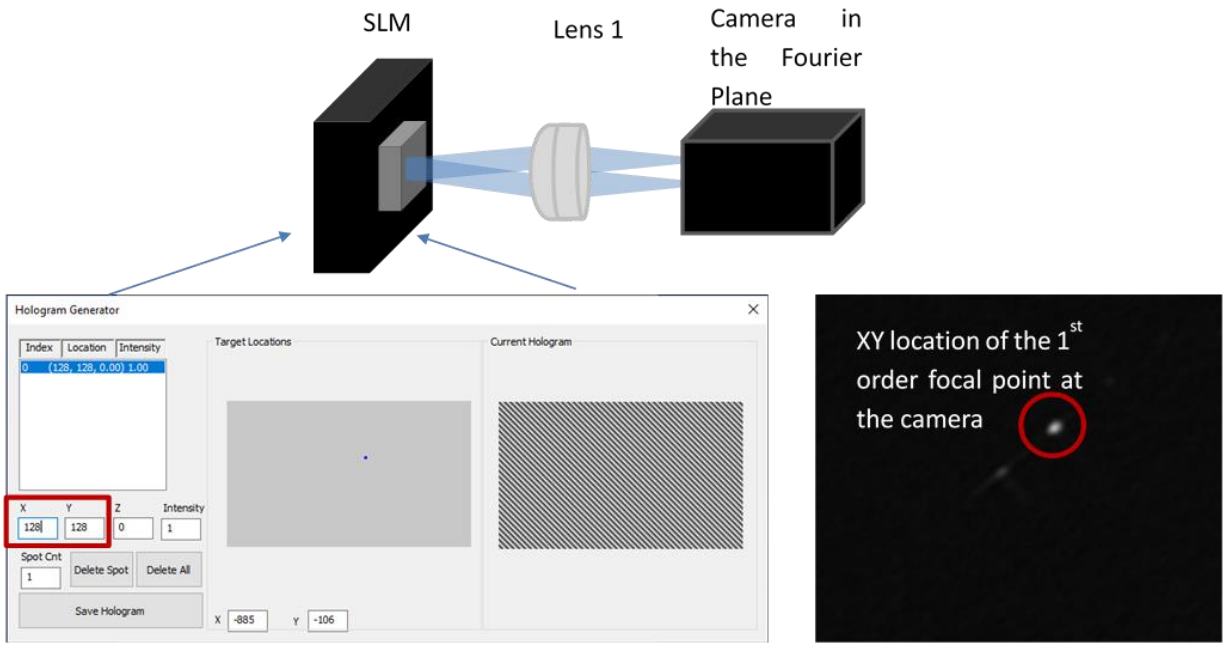


Figure 25 Typical diffractive calibration data

*7.5.14 bool Generate_Hologram(unsigned char *Array, float *x_spots, float *y_spots, float *z_spots, float *I_spots, int N_spots, int ApplyAffine);*

This function computes a hologram to create one or more focal points in the Fourier Plane using a Weighted Gerchberg Saxton. The user should supply an array to be filled with image data by the GPU, an array of x,y,z locations and corresponding focal point intensities, the number of focal points that the user is supplying, and indicate if the affine transformation should be applied. In order to apply the affine transformation, the user should have already made a call to CalculateAffinePolynomials to generate the required mapping of SLM locations to camera coordinates.

7.5.15 void Destruct_HologramGenerator();

This must be called when done using the GPU to compute holograms.

7.5.16 int Initalize_GerchbergSaxton();

For some applications customers would not like to make a hologram to generate focal point locations but would instead like to make a hologram to create an image in the Fourier Plane. There are many algorithms to support calculation of holograms for images, one of which is a Gerchberg Saxton. This function must be called once to initialize the Gerchberg Saxton hologram generator.

*7.5.17 int GerchbergSaxton(unsigned char *Phase, unsigned char* InputImg, int width, int height, int Iterations);*

This function computes a hologram of an image using a Gerchberg Saxton. The user should supply an array to be filled with image data by the GPU, an array containing the

desired image to be created in the Fourier Plane, the height and width of the SLM, and the number of iterations used in the hologram computation.

7.5.18 void Destruct_GerchbergSaxton();

This must be called when done using the GPU to compute holograms of images.

7.5.19 bool Initalize_RegionalLUT(int width, int height);

A look up table (LUT) calibration to linearize the optical response of the liquid crystal to applied voltage. A regional LUT is generated by characterizing the phase vs. voltage response regionally (see section 6) such that the calibration is spatially correct across the SLM. This calibration relies on the GPU to take a table of discreet measurements and generate a pixel by pixel phase calibration to be applied to the image. This function opens communication with the GPU, and specifies the height and width of the images that will be processed through the calibration such that buffers can be allocated.

7.5.20 bool Load_RegionalLUT(const char const RegionalLUTPath, float* Max, float* Min);*

Using the directions in Section 6 a regional calibration can be generated. This function loads the static table of discreet measurements to the GPU such that the pixel by pixel calibration map can be calculated. The function takes a path to the regional LUT file, and fills the contents of empty variables (Max and Min) with the maximum and minimum graylevels found in the regional LUT. The regional LUT is applied in software, and the global LUT applied in hardware should be defined by the user as a linear LUT that ranges from the minimum graylevel in the regional LUT to the maximum graylevel in the regional LUT.

*7.5.21 bool Apply_RegionalLUT(unsigned char *Array);*

This function processes images through the pixel by pixel phase calibration. This must be called prior to loading the image to the hardware.

7.5.22 void Destruct_RegionalLUT();

This must be called when done using the GPU to apply regional phase calibrations to images.

7.5.23 void Mask_Image(unsigned char Array, int width, int height, int region, int NumRegions);*

This function is used in the calibration of a regional LUT to mask off all of an image to 0 except the specified region. For example, if NumRegions is 64, and region is 0, then the SLM is divided into 8x8 regions, and all image data outside region 0 is set to 0.

7.5.24 *bool SetBESTConstants(int FocalLength, float BeamDiameter, float Wavelength, float SLMpitch, int SLMNumPixels, float ObjNA, float ObjMag, float ObjRefInd, float TubeLength, float RelayMag);*

This function sets optical constants that are used in the BEST Bessel beam calculation. Focal Length is the focal length of the lens between the SLM and the amplitude mask. Beam diameter is the diameter of the beam incident on the SLM. Wavelength is the wavelength of excitation. SLM pitch is the pixel pitch of the SLM model used. SLM Num Pixels is the number of pixels along the short axis of the SLM. The objective specifications are the NA, Magnification, and refractive index (of air, water immersion if applicable, or oil immersion if applicable). The tube length depends on the objective manufacturer: Olympus objectives should use a tube length of 180, Nikon of 200, Zeiss of 165, and Leica of 200. The relay magnification is the total magnification from the amplitude mask to the objective.

7.5.25 *bool GetBESTAmplitudeMask(float* AmplitudeY, float* Peaks, int* PeaksIndex, float Period);*

This function calculates the lateral intensity profile of the excitation at the amplitude mask. The function takes an empty array. The elements in the array can be computed as follows:

Objective Pupil Radius = (Objective NA*Tube Length) / objective magnification
Center Mask = (effective NA * Objective Pupil Radius) / Total Optical Relay Magnification
Amplitude Y Array Length = (Center Mask * 1000 * 2) + 1

The Peaks and Peaks index arrays are arrays with five elements that are used to note the value of the peaks and nulls along the axial intensity profile, as well as the index in the array that contains the peaks and nulls. The indices are noted as follows:

0 – Inner Peak
1 – Inner Null
2 – Center Peak
3 – Outer Null
4 – Outer Peak

If using Case I as described in Rongwen et. al., then the amplitude mask rings should cut on the inner peak and the outer peak. If using Case II, then the amplitude mask rings should cut on the inner null and the outer null. For more information please read the following paper and corresponding supplemental information.

Lu, Rongwen, Wenzhi Sun, Yajie Liang, Aaron Kerlin, Jens Bierfeld, Johannes D. Seelig, Daniel E. Wilson et al. "Video-rate volumetric functional imaging of the brain at synaptic resolution." *Nature neuroscience* 20, no. 4 (2017): 620.

The last parameter passed to this function is the period of concentric ring pattern, which is a floating point number. This is calculated as:

Period = Focal Length Lens 1 * wavelength / SLM pixel pitch / Center Mask;

7.5.26 bool GetBESTAxialPSF(double axialAmplitude, float* Intensity, float Period, float OuterDiameter, float InnerDiameter);*

This function calculates the axial PSF of the Bessel excitation beam. The first parameter is an empty array that will be filled with the axial intensity profile. The length of this array is calculated as:

Axial Amplitude Length = Objective Pupil Radius / Total Optical Relay Mag. * 1000 * 2);

The Intensity array is also an empty array that is filled with the lateral intensity of the excitation at the objective. The length of this array is the same as the Axial Amplitude Length. Last the function takes the period of the concentric ring pattern (see 5.5.21) and the diameter of the amplitude mask cutoff.

If using Case I
Outer Diameter = 2.0 * Peaks[OuterPeak]; //mm, outer diameter of annular ring in the MASK
Inner Diameter = 2.0 * Peaks[InnerPeak];

If using Case II
Outer Diameter = 2.0 * Peaks[OuterNull]; //mm, outer diameter of annular ring in the MASK
Inner Diameter = 2.0 * Peaks[InnerNull];

7.5.27 void Generate_BESTRings(unsigned char Array, int width, int height, int centerX, int centerY, float S);*

The function takes an empty array that will be filled with concentric ring image data, the width and height of the SLM, and x and y center of the SLM image (generally half the SLM height and half the SLM width), and a floating point period for the concentric ring period.

8 Generating a Custom LUT Calibration

8.1 Background

All Meadowlark LCOS spatial light modulators ship with a custom calibration file called a LUT file (look-up table). The LUT files are generated with one of the standard laser wavelengths available at Meadowlark Optics (405 nm, 532 nm, 635 nm, 785 nm, 1064 nm, or 1550 nm). A LUT file is required because optical response of liquid crystal (LC) to applied voltage is nonlinear. If your optical system uses a different wavelength, or a different angle of incidence, or a different incident power then generating a calibration in your optical system is highly recommended. This document will walk you through the background of the process we recommend for calibrating the SLM and the steps you should follow to generate your own calibration.

8.2 Introduction to the Diffractive Calibration Method

The method Meadowlark utilizes to calibrate measures the optical response of the SLM as a function of applied voltage using diffraction. By tuning the frequency of the diffraction pattern used in the calibration to match the frequency of patterns in your experiments the calibration can be optimized for your experimental conditions. For example, if utilizing high frequency holograms it is recommended that a high frequency diffraction pattern be used to generate the calibration. The provided post processing software will read in the raw measurements you collect and map input graylevels to output phase. The output of the post processing software is either a regional or a global LUT file that is used to linearize the phase response between 0 and ‘n’π (in most cases 0 and 2π).

Figure 26 shows a typical optical setup for diffractive calibration. If a global calibration is being generated, then the SLM should be illuminated with a beam that covers the active area of the SLM without extending beyond the active area. If a regional calibration is being generated, then it is recommended that the incident illumination be as uniform as possible without reducing the regional intensity to the point that the 1st order diffraction pattern is in the noise. A lens should be selected with a long focal length (~100 mm) to provide enough separation between the diffraction orders to align the detector. An iris may be used in front of the detector to block detection of light from unwanted diffraction orders.

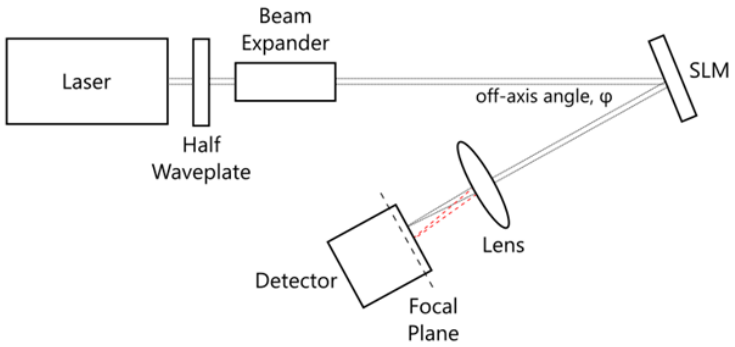


Figure 26 Typical optical setup for diffractive calibration

In the SDK folder Meadowlark Optics provides example programs written in C++, Matlab, and LabVIEW, which are named PCIeDiffractiveTest (.cpp, .m, or .vi). This example program is used to collect the raw measurements required by the post processing program (DiffractiveLUT.exe) to generate a LUT. The example program will generate and load a series of binary phase grating images to the SLM. These images can either be stripes or checkerboards with one value held constant (typically 0) and the other varying from 0 to the maximum greyscale level. We recommend using stripes if measuring first order intensity to ease alignment. The width of the diffraction grating pattern written to the SLM during this test should be sufficiently small to clearly separate the 0^{th} and 1^{st} order. We typically use, and recommend, either 4 or 8 pixels per stripe. The customer should use an analog input board to measure the intensity of either the zeroth or first order diffraction spot as each pattern is loaded to the SLM.

Figure 27 shows typical raw measurements collected utilizing the diffractive method. In this case the 0^{th} order was measured as the diffraction patters were loaded to the SLM. Collection the 0^{th} order is acceptable for a global calibration, but a regional calibration requires collection of the 1^{st} order. The data has been normalized to the maximum and minimum values. Greyscale Delta is the difference between the binary grating values. 0% is a fully black image (greyscale value 0), whereas 100% is a fully black and fully white grating (greyscale value 0 and greyscale value 255). The dotted lines represent the phase shift locations for π and 2π. The first null or peak (depending on if the 0^{th} order, or 1^{st} order is collected) is π and the second null or peak is 2π.

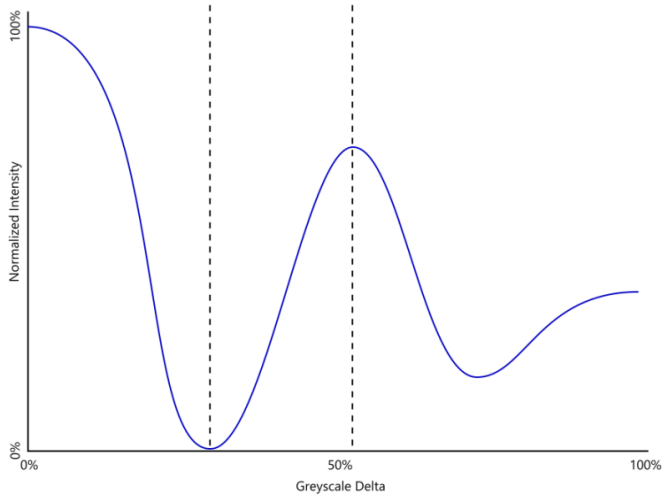


Figure 27 Typical diffractive calibration data

Figure 28 shows a simulated Fourier plane generated by the SLM when using the diffractive calibration method is used. If writing binary phase grating stripes to the SLM a $\pm 1^{st}$ order will be generated about the 0^{th} order. In order to generate a LUT, the customer will use the Meadowlark Optics provided example programs to load diffraction patterns to the SLM, and read from an analog input board to store the 1^{st} order intensity (or 0^{th} order intensity) measurements and save the files in an appropriate format for generating a calibration. The contents of the collected files

will look similar to that shown in Figure 2. After collecting the measurements, DiffractiveLUT.exe can be used to generate the calibration.

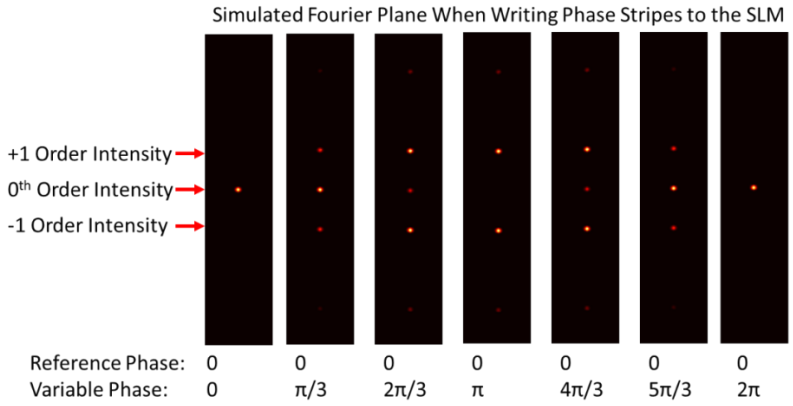


Figure 28 Simulated Fourier Plane with stripe patterns

8.3 Collecting the Raw Measurements

Meadowlark optics provides example programs written in C++, Matlab, and LabVIEW that can be used to collect the raw measurements required to generate a LUT. The example programs are named PCIeDiffractiveTest (.cpp, .m, or .vi). The order of operations in the examples is as follows:

1. Open communication to the SLM
2. Load a linear LUT to the SLM – this effectively disables the LUT such that the raw optical response of the liquid crystal to applied voltage can be measured
3. Loop through the number of regions in the calibration. A global LUT will use one region, or a regional LUT will use 64 regions. The measurements of regional data are built into a pixel by pixel 3D map to spatially linearize the phase response of the SLM from 0 – 'n'π.
4. Loop through the 256 diffraction patterns. First, the example generates a stripe image. Next, if generating a regional calibration, the image is masked off to graylevel 0 except the current region of interest. This masked image is written to the SLM. A delay allows the liquid crystal to settle into the new phase pattern.
5. After the phase pattern is written to the SLM, **you must fill in your own code to read from your particular analog input board.**
6. After each region is measured, the raw intensity vs. grayscale measurements are stored in Raw[region].csv to be post processed by DiffractiveLUT.exe. For example, if generating a global LUT there will be one file generated called Raw0.csv, and if generating a regional LUT there will be 64 files named Raw0.csv to Raw63.csv.

8.4 Generating a LUT from Raw Measurements

The raw measurements are post processed using DiffractiveLUT.exe. A screen capture of the program is shown for a global LUT in Figure 29, and for a regional LUT in Figure 30. Most of the controls are disabled when the program is first opened. The user can click on the Global LUT checkbox to indicate if they are generating a LUT with one region, or with 64 regions. After setting

the number of regions by setting the state of the checkbox, the user can browse to a folder containing the Raw*.csv files.

After the raw files have been read in the software will convert intensity to phase, and apply a curve fit to the phase versus grayscale trace.

There are several controls on the GUI that the user should be aware of:

1. **Global LUT** Check if calibrating the SLM as a single region, and do not check if calibrating regionally.
2. **Phase** Check to generate a phase calibration, or uncheck to generate an amplitude calibration. For an amplitude calibration the input and output polarizers must be oriented such that the intensity starts at mid-scale intensity and increases to a peak. The calibration will be made from the first peak to the first null.
3. **In Bit Depth and Output Bit Depth** Please reference the table below to determine the appropriate values for your SLM model

SLM Model	Input Bit Depth	Output Bit Depth
512, 8 bit PCIe	8	8
1920, 12 bit PCIe	8	12

4. **Raw Button** Clicking this button will allow you to browse to Raw*.csv files. The software will automatically convert the intensity measurements to phase, apply a curve fit, and update the graph. This will also generate excel csv files in the folder containing the raw files. The first file is a global normalization of the raw measurements. The second file is a normalization from 0 to 'n'π. The third shows the unwrapped phase vs. grayscale measurements. The fourth is a curve fit of the phase vs. grayscale data. The fifth is the normalized LUT where 0 to 1 corresponds to 0 to 'n'π. Seeing the output of each step of the LUT generation process allows you to see if the program failed to properly convert the raw measurements into a LUT. This can also be checked visually using the plots on the GUI. If the raw measurements are too noisy, the algorithm could fail to properly convert the intensity data to a calibration.
5. **Calibrated Button (optional)** Clicking this button will allow you to browse to Raw*_Calibrated.csv files. This is a way to validate your calibration. First, you can use the provided example programs to generate raw measurements with a linear LUT applied to the SLM and post process the raw files with DiffractiveLUT.exe to produce a calibration. Next, you can repeat the process with the calibrated LUT applied instead of using the linear LUT. Now the output files should be named Raw[region]_Calibrated.csv. By browsing to the calibrated raw files you can generate regional or global plots and excel files to validate that the calibrated phase vs. grayscale response is linear.
6. **Zero Order Checkbox** The zero order checkbox is used to tell the software if the intensity measurements were collected off the 0^{th} or 1^{st} order.
7. **Num Waves** Using the Num Waves editbox the user can dynamically set the number of waves the calibration linearizes over: 0.5 waves = 0 to 1π, 1 wave = 0 to 2π, 1.5 waves =

0 to 3π , and 2 waves = 0 to 4π. After editing Num Waves the plot of the intensity to phase measurement and the curve fit will automatically update.

8. **Moving Average** If there is noise in the raw data it may be possible to reduce the noise by applying a moving average. Setting Moving Average to 0 will disable the moving average. Setting Moving Average to 1 will average one data point before, the current data point, and one data point after for each data point in each Raw*.csv file.
9. **SLM Width and SLM Height** Use the resolution of your SLM model.
10. **SLM Serial Number** This is used for naming the output calibration.
11. **Wavelength** This is also used for naming the output calibration.
12. **Save** When this button is clicked the Normalized LUT data is scaled to the appropriate output bit depth, and the user is prompted for a filename and location to save to. A global LUT is saved as a *.LUT, a regionals LUT is saved as a *.txt. Both file formats are accepted by Blink and the SDK functions.

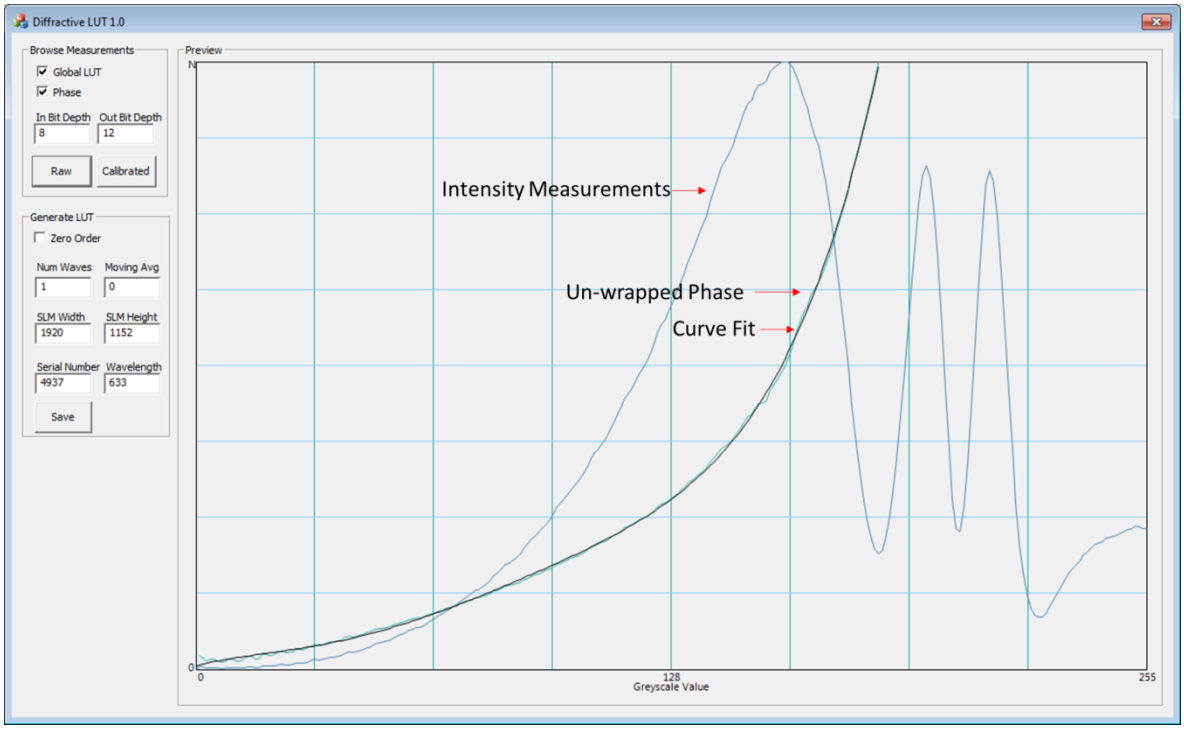


Figure 29 Global LUT: After the user browses for the raw measurement, the software will plot the intensity, the un-wrapped phase, and the curve fit. The user can scale the number of waves used in the calibration, and save the calibration.

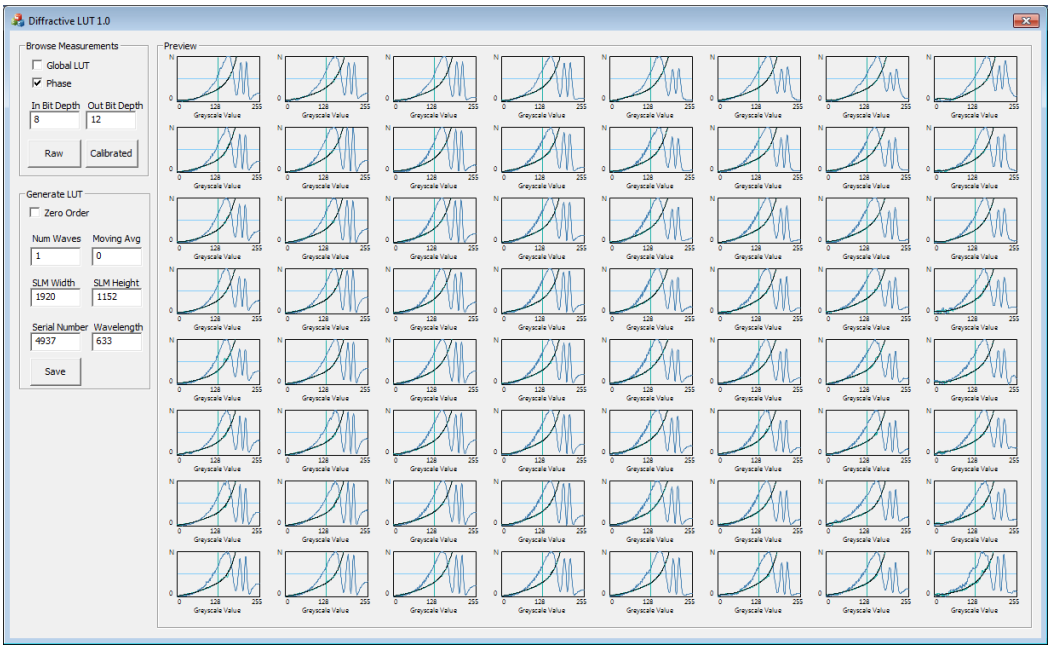


Figure 30 Regional LUT: After the user browses for the folder of raw measurements, the software will plot the intensity of each region as well as the corresponding regional un-wrapped phase, and the curve fit. The user can scale the number of waves used in the calibration, and save the calibration

8.5 Post Calibration Validation

If desired, the process can be repeated while applying either the regional or global calibrated LUT as opposed to using the linear LUT. The calibrated phase response should be linear over 'n' waves as shown in Figure 31.

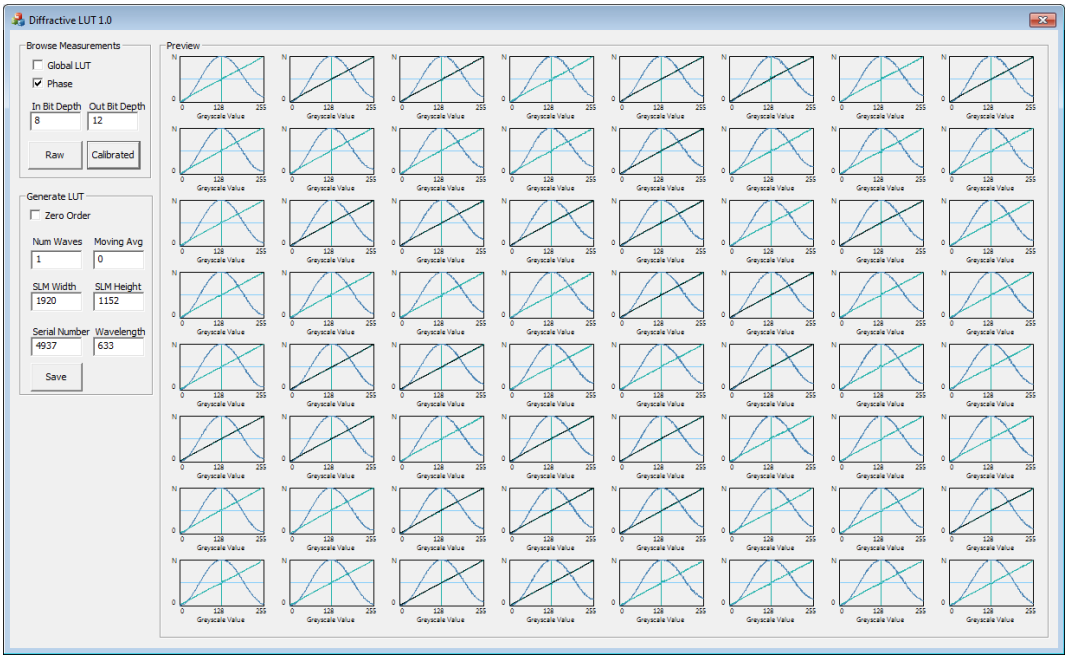


Figure 31 Post calibration regional LUT validation.

9 Mechanical Housing

The SLM Housing has vertical and horizontal micrometers for adjusting the tip and tilt up to ± 3°. The housing also allows the user to rotate the SLM about the optical axis by ~15°. A set screw (located between the tip/tilt knobs on the side of the SLM housing) locks the rotation about the optical axis.

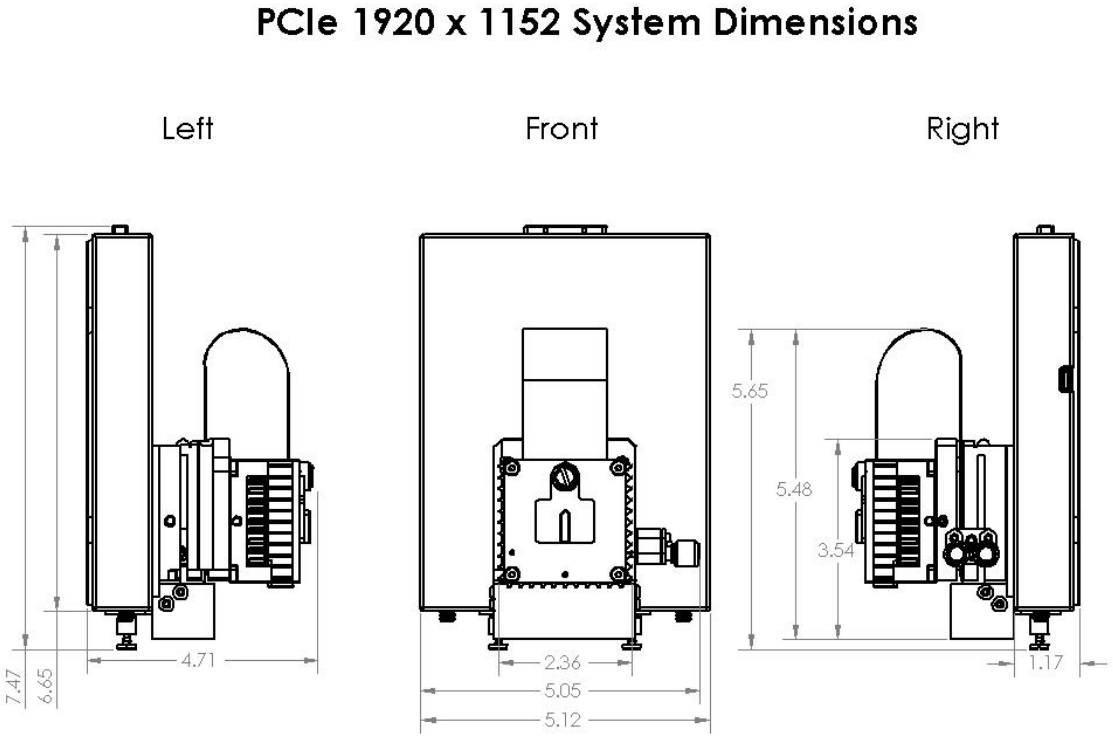


Figure 32 Outline drawing showing front and side views of 1920 SLM Optical Head. Dimensions are in inches.

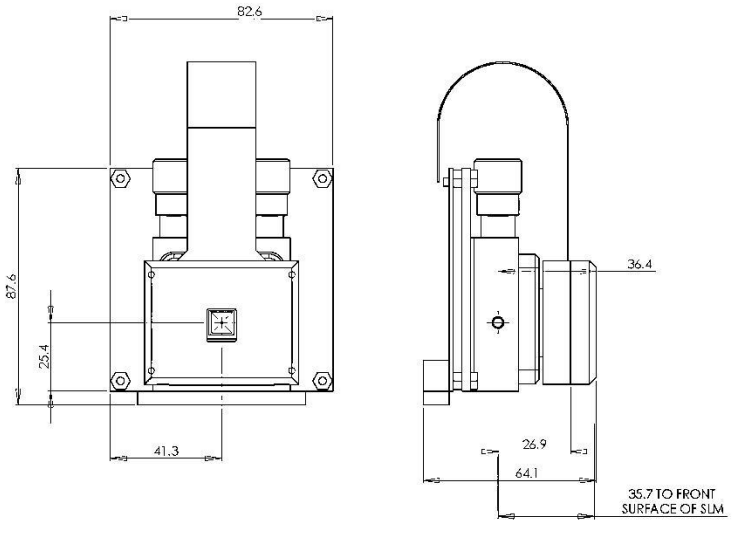


Figure 33 Front and side views of 512 SLM Optical Head. Dimensions are in millimeters.

10 SLM care and cleaning

CAUTION

Failure to follow the recommendations included in this document may violate your warranty. Always ensure all personnel involved in the handling of your SLMs have appropriate tools, equipment, and training prior to cleaning any SLM.

Never use acetone to clean your SLM. Acetone will cause irreparable damage to your SLM.

Never use pressurized air or nitrogen to clean your SLM. Pressurized air or nitrogen will destroy the bond wires that provide the electrical interface to the SLM.

The SLM is shipped with rotating aperture cover covering the SLM. This aperture, or a similar aperture, should be kept in place while the SLM is not is use to reduce the buildup of dust and debris on the SLM coverglass. When removing the aperture cover, or while the SLM coverglass is exposed, care should be taken to avoid any contact with the coverglass.

If dust or debris does contaminate the coverglass, gently remove them with a methanol soaked lens tissue. Take a small lens tissue or cleanroom wipe and soak a corner of it with methanol. Let the methanol evaporate until the lens tissue is just damp then gently wipe off the particles with the corner of the wipe. For more stubborn debris the methanol drag can be used. Fold a lens tissue several times until it is just small enough to fit inside of the SLM aperture. Soak the folded edge with methanol and let it evaporate until just damp. Line up the folded edge of the lens tissue with the edge of the glass, and drag the tissue straight across the glass. Never wipe the glass more than once with the same lens tissue as it will merely transfer dirt back onto the coverglass. Repeat as necessary.

If there is any dirt or contamination that cannot be removed, please contact the factory for additional cleaning instructions.

11 Troubleshooting

11.1 Image appears incorrect

When loading images to the SLM, make sure the image files are 1920 x 1152 8-bit bitmap images for the 1920 SLM, or 512x512 8 bit bitmap images for the 512 SLM. If the image dimensions are incorrect, it will not be displayed properly on the SLM.

11.2 LEDs on the 1920x1152 PCIe hardware are blinking

There is dual color LED on the front panel of the PCIe hardware, which indicates various states:

LED behaviour	**State**
RED	During initial power up, if the red LED is on then external power is present but one of the internal regulators is faulty.
AMBER	FPGA not configured
RED blinking	Temperature alarm, FPGA temperature is above 90C. *Note: the alarm is removed when FPGA cools down to 85C*
AMBER blinking	LCOS not in sync, for example LCOS disconnected or faulty
GREEN blinking	PCIe not in sync, for example PCIe cable disconnected
GREEN	Normal operation, PCIe is in sync, LCOS is in sync

Thank you for purchasing a Meadowlark Optics Spatial Light Modulator.

For additional product and company information, please contact:

Meadowlark Optics, Inc.
5964 Iris Parkway
P.O. Box 1000
Frederick, CO 80530

Telephone: 303-833-4333
Fax: 303-833-4335
Website: www.meadowlark.com

Please feel free to contact us with any questions you may have as well as to leave feedback about your device.

For questions regarding customer support for Meadowlark SLM products, please contact us by telephone or by e-mailing slmsupport@meadowlark.com

For questions regarding purchasing and pricing of additional Meadowlark SLM products, please contact us by telephone or by e-mailing sales@meadowlark.com